Universitat de Lleida

TREBALL FINAL DE GRAU

ESCOLA POLITÈCNICA SUPERIOR
UNIVERSITAT DE LLEIDA
INSPIRING THE FUTURE

Estudiant: Alex Satour Perucho

Titulació: Grau en Enginyeria Informàtica

Títol de Treball Final de Grau: **Ethnographic analysis, design and development of an application to offer an autonomous rehabilitation process to patients.**

Director/a: **Rosa Gil**

Presentació

Mes: Juny

Any: 2019

Abstract

Nowadays, technology is used somehow in every aspect of our lives, improving the tasks we do daily.

Being used to that, it was very strange to see that when you have to go through a rehabilitation process, most centres do not use the capabilities offered by current technology. Forgetting that almost all population is used to interact with smartphones and mobile applications.

This project is going to study the needs required and an application that provides all the functionalities is going to be designed and developed. As a result, this project will provide a new way to improve rehabilitation processes.

Throughout this document, every necessary phase to build the application mentioned is described. Starting from exploring the solutions offered at the moment, analysing the needs that the users involved have, the main points of designing a prototype and testing it to the implementation of the application that can be used in real life.

This project could radically change the way a rehabilitation process work since it would track users healing process and be able to manage appointments using smartphones that are present in our day to day life.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motives

A few years ago, I had an injury on my elbow that led me into a rehabilitation process. Once I was there, I was able to see the necessities that the medical crew had and I started to think what could I do to improve both the effectiveness of their work and the rehabilitation process of patients.
It wasn't until I took a subject that explained the main points to guarantee a good interaction between human and digital systems that I came to the conclusion that I could try to improve the weak points seen throughout my rehabilitation process. In addition, they could be a good target be set in order to study all the processes that go into the process of developing an application that was what was really interesting to me.

1.2 Goals

The main goal of this project is to study all the different phases that are necessary to build an application, just as if it was done for a real application that would be distributed to final users.
This would include the interaction with experts on other areas all throughout the development in order to be able to have the necessary information to make the proper decisions and assure that everything is done for a reason.
In addition, another target that is pursued is to be able to learn about the area studied in each phase of the project. Mainly on the stages of designing and implementing the application, since these are the ones that are wider and leave more room for experimentation.
Furthermore, the ability to be able to structure all the tasks and time them in order to be as effective as possible is another huge goal to be achieved in this project.

1.3 Stages

This project will be developed following the Agile methodology[1]. It was introduced by Dr. W. Royce in a paper published in 1970, this method assists teams in responding to the unpredictability of constructing software. It uses incremental, iterative work sequences that are commonly known as sprints.
It is based on sprints, they are periods of time allocated for a particular phase of a project. Sprints are considered to be complete when the time period expires. There may be disagreements among the members of the team as to whether or not the development is satisfactory, however, there will be no more work on that particular phase of the project. The remaining phases of the project will continue to develop within their respective time frames.
So, at the very beginning of the project, almost all sprints were defined, establishing their respective time frames in order to be able to divide the time available among all the tasks.
On the first phase, it was very important to know the state of technologies that will be used by this project as well as the solutions that are offered at the moment that use some kind of technology to improve a rehabilitation process. All of the information described can be found in section 2 named state of art. Then, the idea was to gather as much information necessary to properly develop the project and to set its requirements. This phase is called Analysis and is described in section 3. First and, as exposed in section 3.1, the phase of software requirements started. This stage consists of gathering as much information about the user expectations on a product as well as the requirements it will have and the needs it might solve. This stage is very useful since it will lead us to good results because we will have knowledge in almost every area covered by the project.
Obviously, the main point was to gather all the possible information obtaining it both from the patients and the medical staff that works for them. In order to get the information related to the patients, a survey was created that provided information about their knowledge about the usage of applications and smartphones and if they had a disability that affected their interaction with electronic devices.

This way of developing applications is called User Centred Design [2], it is described as an iterative design process in which designers focus on the users and their needs in each phase of the design process. UCD calls for involving users throughout the design process via a variety of research and design techniques so as to create highly usable and accessible products for them.
In this case, many interviews with physiotherapists and rehabilitation doctors were conducted to study their needs, the profiles that they usually have on a rehabilitation centre and acquire some knowledge about the main aspects and concepts that are used and seen. The process and results of the gathering of the information mentioned above are explained in section 3.3 named information gathering.
Once all the information was gathered, it was time to start designing the application taking into account all the knowledge obtained in previous states. Two prototypes had to be designed since there will be two versions of the application, one for the patients and another for the medical crew members. Of course, these prototypes will share many screens since they need similar functionalities in order to solve the needs of these two types of users. All the information about the development of this stage such as the tools used or the process of designing the prototypes themselves can be found in section 4 named design.
When a first prototype was designed, it was tested with some users to see their reactions and the way they interacted with the system, in order to spot all the weak points and improve them. By doing this, we can assure that the design used is optimal and will save us costs in future parts of the project since the sooner an error is fixed the cheaper it is. The results and development of the testing process is described in section 4.4 named tests with users.
When the prototype was tested, it was time to start coding and developing the application that would satisfy the requirements established on the initial phases of the project. Many technologies had to be explored in order to find the optimal ones to use for this project. All the information form the tools and technologies used can be found in section 5 named development. Furthermore, in this section is also explained the main points of the development process such as how the crucial functionalities were implemented or the reasons behind some decisions that had to be taken.
Lastly, the conclusions of the development of this project can be found in section 6. In addition, as in any project with limited time for developing, there are some points in which the project could be improved but have not been implemented due to timing issues or to the current state of the tools used. All of them are explained in section 7 named future work.

2 State of the art

When the survey phase was finished. The results were gathered in order to study the best options to develop the application. The results didn't provide a concise answer, because even though most of the people used an Android device, a big part still used some kind of iOs device.
As a result, many other points were studied in order to decide between these two operating systems, leading to using Android as the target platform. This decision was taken for many reasons but mainly because if the system was to be implemented in a real centre, android devices are much cheaper. In addition, testing the application and the distribution of a non-final version to test in different devices is much easier.
In this section, the main points about Android are going to be explained in order to be able to understand better the state of the operating system as well as its strengths and weak points in order to be aware of the capabilities offered and use them if necessary.
In addition, the main projects that currently explore the usage of technologies in the rehabilitation process are going to be described in order to know what has been done until now and how can this project contribute to improving the healing process of patients.

2.1 Operating System

The operating system chosen to develop this project for is Android. The reasons why are explained later on in this document but it is mainly because most users already own some kind of Android device and, in addition, it is a very open operating system that will ease the tests of the application in real devices. Furthermore, Android devices can be really inexpensive, leading to a reduction of costs if some gadgets have to be acquired.
Android[3] is a mobile operating system developed by Google. It is based on a modified version of the Linux kernel and other open source software and is designed primarily for touchscreen mobile devices such as smartphones and tablets. In addition, Google has further developed Android TV for televisions, Android Auto for cars, and Wear OS for wrist watches, each with a specialised user interface. Variants of Android are also used on game consoles, digital cameras, PCs and other electronics.
Initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, with the first commercial Android device launched in September 2008. The operating system has since gone through multiple major releases, with the current version being 9 "Pie", released in August 2018. The core Android source code is known as the Android Open Source Project (AOSP) and is primarily licensed under the Apache License. Android has been the best-selling OS worldwide on smartphones since 2011 and on tablets since 2013. As of May 2017, it has over two billion monthly active users, the largest installed base of any operating system, and as of December 2018, the Google Play store features over 2.6 million applications.

2.1.1 Platform Architecture

Android is an open source, Linux-based software stack created for a wide array of devices and form factors. Its structure can be observed in the following figure:

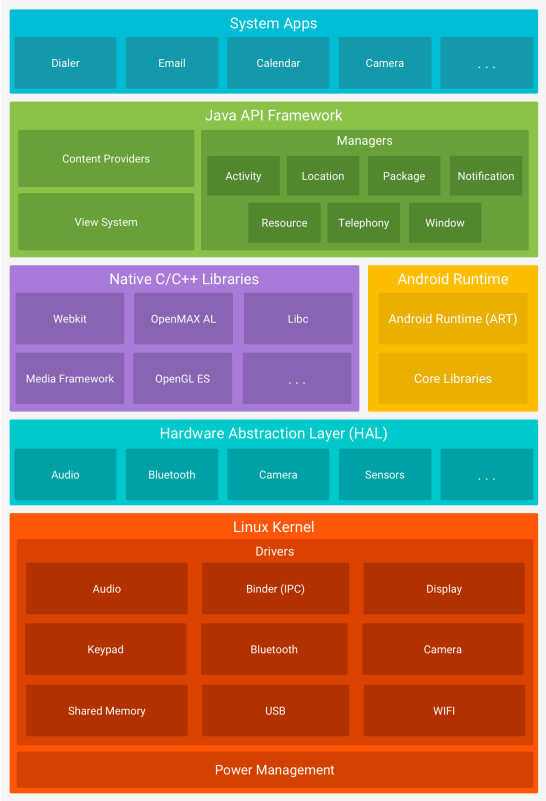

Figure 1: Android's Architecture[4]

As seen in the figure, the architecture[7] has many layers. The foundation of the Android platform is the Linux kernel. For example, the Android Runtime (ART) relies on the Linux kernel for underlying functionalities such as threading and low-level memory management. Using a Linux kernel allows Android to take advantage of key security features and allows device manufacturers to develop hardware drivers for a well-known kernel.
The hardware abstraction layer (HAL) provides standard interfaces that expose device hardware capabilities to the higher-level Java API framework. The HAL consists of multiple library modules, each of which implements an interface for a specific type of hardware components, such as the camera or Bluetooth module. When a framework API makes a call to access device hardware, the Android system loads the library module for that hardware component.
For devices running Android version 5.0 (API level 21) or higher, each application runs in its own process and with its own instance of the Android Runtime (ART). ART is written to run multiple virtual machines on low-memory devices by executing DEX files, a bytecode format designed specially for Android that is optimised for minimal memory footprint. Build toolchains, such as Jack, compile Java sources into DEX bytecode, which can run on the Android platform.
Many core Android system components and services, such as ART and HAL, are built from native code that requires native libraries written in C and C++. The Android platform provides Java framework APIs to expose the functionality of some of these native libraries to applications.
The entire feature-set of the Android OS is available through APIs ((Application Programming Interface) written in the Java language. These APIs form the building blocks needed to create Android applications by simplifying the reuse of core, modular system components and services.
Android comes with a set of core applications for email, SMS messaging, calendars, internet browsing, contacts, and more. Applications included with the platform have no special status among the applications the user chooses to install. So a third-party application can become the user's default web browser, SMS messenger, or even the default keyboard (some exceptions apply, such as the system's Settings application).

2.1.2 System Versions

The version history[5] of the Android mobile operating system began with the public release of the Android beta on November 5, 2007. The first commercial version, Android 1.0, was released on September 23, 2008. Android is continually developed by Google and the Open Handset Alliance, and it has seen a number of updates to its base operating system since the initial release.
All versions have a code name related to deserts or sweets. We can see the information related to all the versions released until now in the figure attached below.

Code name	Version number	Linux kernel version	Initial release date	API level
(No codename)	1	?	September 23, 2008	1
Petit Four	1.1	2.6	February 9, 2009	2
Cupcake	1.5	2.6.27	April 27, 2009	3
Donut	1.6	2.6.29	September 15, 2009	4
Eclair	2.0 – 2.1	2.6.29	October 26, 2009	5 – 7
Froyo	2.2 – 2.2.3	2.6.32	May 20, 2010	8
Gingerbread	2.3 – 2.3.7	2.6.35	December 6, 2010	9 – 10
Honeycomb	3.0 – 3.2.6	2.6.36	February 22, 2011	11 – 13
Ice Cream Sandwich	4.0 – 4.0.4	3.0.1	October 18, 2011	14 – 15
Jelly Bean	4.1 – 4.3.1	3.0.31 to 3.4.39	July 9, 2012	16 – 18
KitKat	4.4 – 4.4.4	3.1	October 31, 2013	19 – 20
Lollipop	5.0 – 5.1.1	3.16	November 12, 2014	21 – 22
Marshmallow	6.0 – 6.0.1	3.18	October 5, 2015	23
Nougat	7.0 – 7.1.2	4.4	August 22, 2016	24 – 25
Oreo	8.0 – 8.1	4.1	August 21, 2017	26 – 27
Pie	9	4.4.107, 4.9.84, and 4.14.42	August 6, 2018	28

Figure 2: Android versions[5]

At the figure above, we can see highlighted in red the versions that are already deprecated, in orange the ones that are old but still receive support and in green the most recent version.
As we can see, the latest version is named Android Pie. It is the ninth major version of the Android operating system. It was first announced by Google on March 7, 2018, and the first developer preview was released on the same day. The second preview, considered beta quality, was released on May 8, 2018. The final beta of Android P (fifth preview, also considered as a "Release Candidate") was released on July 25, 2018. The first official release was released on August 6, 2018.
Its new features[8][9] include:

- **New gesture navigation**: Android has used always a standard three-button navigation bar at the bottom. In Pie, the standard Back, Home, and Recent buttons can be dropped in favour of a new gesture-based navigation system.
- **Adaptive battery and brightness:** Adaptive Battery is an expansion of the Doze feature introduced in Android 6 Marshmallow. Doze put applications that users were not using into a “deep sleep” to prevent them from wasting battery. Now, Adaptive Battery goes further by learning about the applications and services used most often, then adjusting what is not used as much to use less battery.
- **Improved notifications**: Android will provide smart replies to certain notifications to let user respond quicker.
- **Native notch support**: Many phones released lately have a notch on their screen and Pie supports it. Leading into better management of the screen space.
- **Digital Well-being**: It is a series of tools designed to inform users about how they actually use their phones.

One of its weakest points is how many different manufacturers create devices that run Android. This leads to having a big dependency on them to update their devices since Android has to be adapted to be installed on every device because each one has its own components. As a consequence, the introduction of software updates on the market is very slow and there is a big fragmentation as seen on the figure attached below.

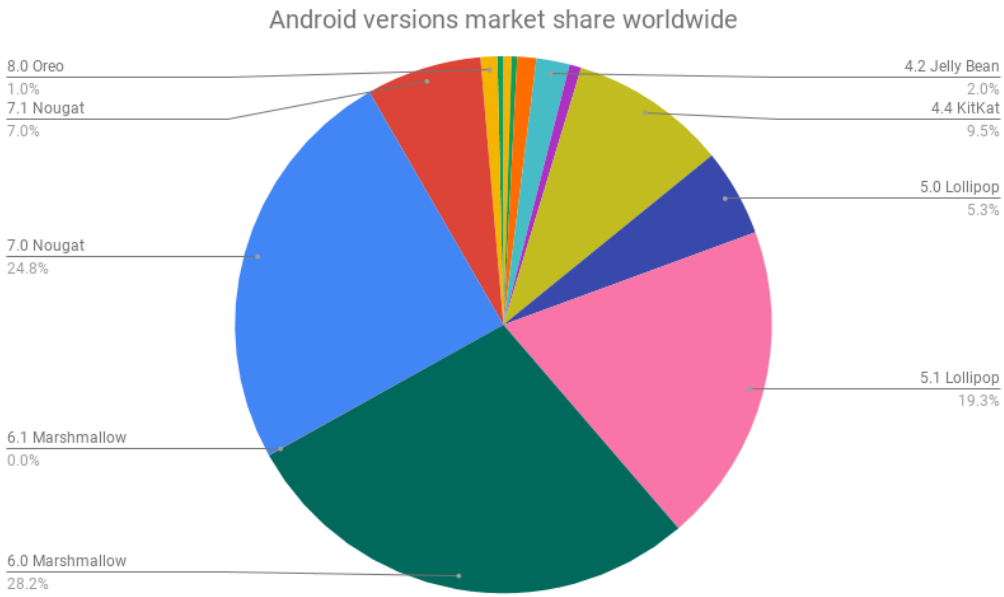

Figure 3: Market share of every android version in February 2019.[6]

As we can see, there are devices running very old versions of Android such as KitKat or JellyBean. Therefore, the choice of the minimum Android version needed to run the application is really crucial.
For this project, the minimum version required will be Android 5.0 lollipop, since it is the middle point between having as many new functionalities and still being able to reach as many users as possible. More specifically, setting this version as the minimum would mean that the application could be used in approximately 85.8% of devices that are on the market right now.

2.2 Existing proposals

A few approaches to this idea of improving the rehabilitation process using devices have been implemented although none of them pursues the idea behind this project. This is because all of them are more related to be able to track and obtain precise information about the rehabilitation process and the evolution of an injury. In addition, one is specifically designed to only treat and track knee injuries.
As a consequence, this would be the first system to manage the exercises that patients should do and to track their feedback, at least as far as researched.

Toyra

Toyra [10] is a product to track the rehabilitation of the upper part of the body using a virtual reality system and motion capture in real time. This system was designed by the multinational INDRA in cooperation with the *Hospital Nacional de Parapléjicos de Toledo* and the foundation *Rafael del Pino*.
The movements are captured by sensors (using various motion capture technologies), then they send to the system its location and position, thus being able to recreate the movement that the patient performs. With the information received Toyra evaluates, registers and analyses the results obtained by a patient during the execution of the Therapy.

Figure 4: Patient using Toyra's system to heal an injury[11]

As said before, this project could be complementary to the application that is being developed since the functionalities offered could lead into a better tracking of the evolution of injuries as well as offering a way to do the exercises in a more interactive, visual and precise manner.

DyCare

DyCare [12] is a company that designs portable solutions that help on the rehabilitation process of patients with musculoskeletal problems. They mainly work on two projects: ReHub and Lynx.
ReHub [13] is a digital rehabilitation platform based on medical evidence that delivers effective, personalised home rehabilitation for people who suffer from muscle-skeletal problems. This tool maximises operational aspects and therapeutic effectiveness, promoting continuous communication between the physician, physiotherapist and the patient.

Figure 5: Smart Exercise Kit used on ReHub[15]

The figure attached above is what they call a *Smart Exercise Kit*. This is the equipment that includes the sensors that gather the necessary information. Then this information is analysed and stored by the system in order to record and show every exercise.
Lynx [14] is the other solution given by Dycare aimed at all professionals of physical rehabilitation, consisting of motion sensors and intuitive software that allow a functional assessment of joint disorders. The professional can make an evaluation before, during and after the treatment that facilitates the therapeutic planning according to the needs of the patient.

Rehabitic

Rehabitic [16] is a service that provides telerehabilitation for patients that have had a surgery on their knee to fix a total arthroplasty. It is set up at *Parc de Salut Mar* at the *Hospital de l'Esperança* with the collaboration of *Telefónica*. Their aim has been to take the therapies of rehabilitation at the home of the patient so that they can reduce the displacements and improve the treatments with the maximum clinical guarantees.

Figure 6: Screenshot of a Rehabitic session[17].

This approach has lots of limitations, given that it's target are only people that have had surgery to fix a very specific injury. Nevertheless, and as seen with the projects shown before, their strongest point is the way they can track each exercise and record patient's movements in order to improve the effectiveness of the rehabilitation process.

TANGO: H

The last approach would be Tango: H. It[18] is a motor and cognitive rehabilitation platform aimed at hospitalised children with disabilities. The platform has been developed through a collaboration agreement between ITER and the Interaction, Technology and Education Research Group (i-TED) of La Laguna University's department of systems and automation engineering and computer architecture and sciences.

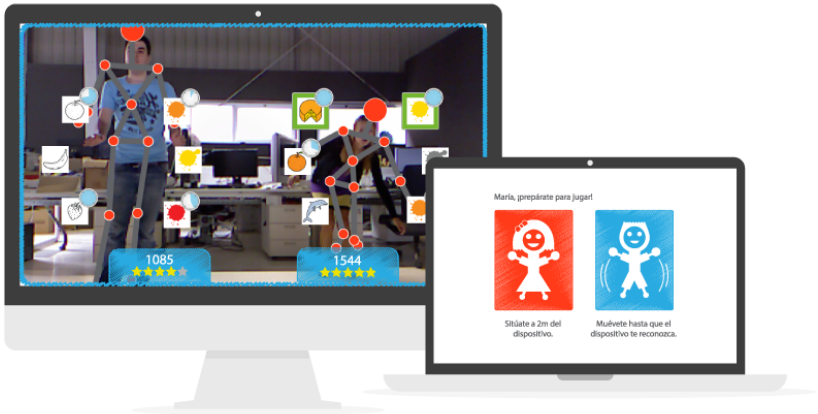

Figure 7: Tango: H's interface[19]

Tango: H bases its technology on Microsoft's Kinect device, which uses an RGB camera and a depth sensor to identify the human body and its environment interacting with its information systems without requiring the physical contact of traditional control systems.

Tango: H has two basic modes of play: single and multiplayer. In turn, in the multiplayer mode, two players can play simultaneously in a collaborative, competitive or sequential way.
The power of Tango: H lies in its activities design module. The designer that comes with the application allows doctors to create exercises that are tailored perfectly to the needs of the players, both cognitive and motor, enabling doctors to create exercises that are educational, recreational, rehabilitational or a mixture of these, with the ability also to almost immediately make quick modifications, if necessary.

EME-REHAB and SIVIRE

EME-REHAB and SIVIRE are two projects[20] developed by Victor M. R. Penichet. EME-REHAB is a software for the Edition and Monitoring of Exercises for Rehabilitation Processes. Its interface can be seen in the figure attached below.

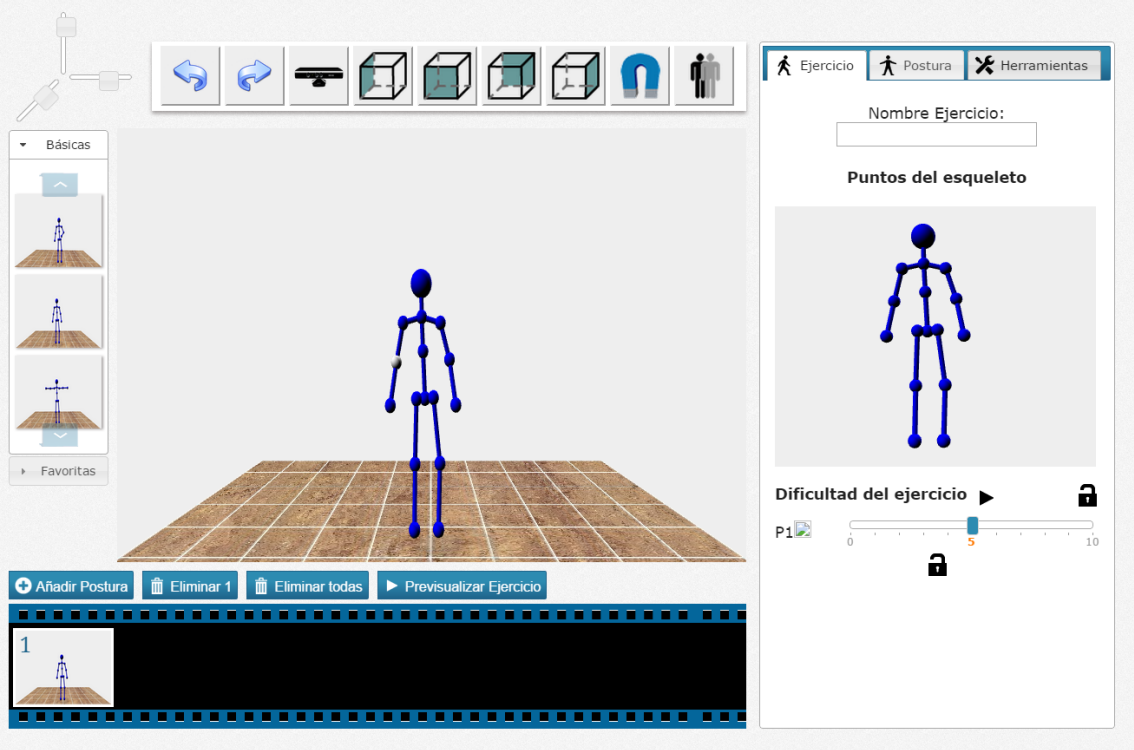

Figure 8: EME-REHAB's interface[21].

Meanwhile SIVIRE is a Virtual System for the Creation of Exercises with Movement-based Interaction for Rehabilitation Processes. A screenshot showing the interface of this project can be seen in the figure 9 attached to the following page.

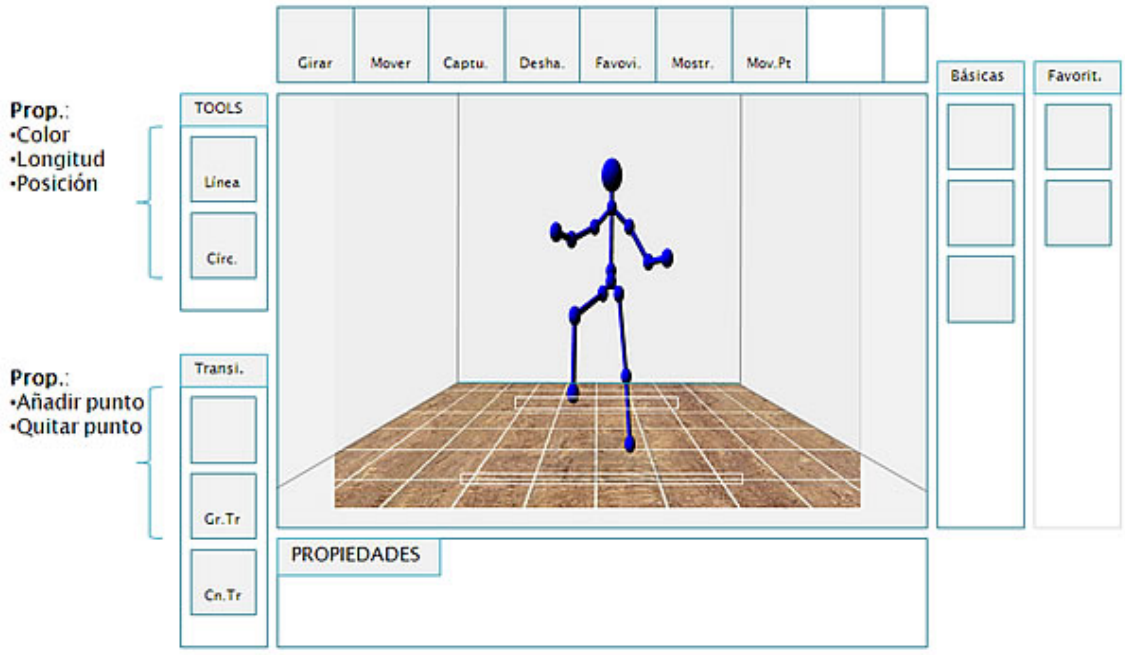

Figure 9: SIVIRE's interface[22].

As we can see, both these projects are completely related to the creation and tracking of exercises to improve the patient's rehabilitation process.

3 Analysis

To find out all the requirements needed to be fulfilled, many tasks were conducted. The first one was creating two kind of surveys, one specific for former rehabilitation patients and another one that could be filled by anyone since we could all be patients if we get an injury at some point of our lives. The main reason why they were done was to get some information about both the knowledge and the ease in interacting with electronic devices and in case of patients, to find out their needs.
In addition, many meetings with doctors and the medical crew of rehabilitation centres were conducted in order to get some information about the technologies they use and the functionalities they might need from this application.

3.1 Requirements

3.1.1 Functional Requirements

Once the first steps were conducted, enough information was gathered to build a list of functional requirements that the system had:

1. Login and Logout
   (a) The application must let both patients and doctors log into the application if the login information is correct.
   (b) The application must let both patients and doctors log out.
   (c) The application must let both patients and doctors change their password.

(d) The application must let an admin create new users to be given to new patients or doctors.

2. Appointments
   (a) The application must let patients see their future appointments.
   (b) The application must let both patients and doctors cancel a future appointment.
   (c) The application must let doctors create appointments.
   (d) The application must let doctors see the appointments of a specific patient.
   (e) The application must let doctors see their future appointments with patients.
3. Rehabilitation Sessions
   (a) The application must let patients start their rehabilitation session.
   (b) The application must let patients leave feedback when an exercise is done.
   (c) The application must let patients see the feedback left on past sessions.
   (d) The application must let doctors create and assign a rehabilitation session.
   (e) The application must let doctors create and remove exercises.
   (f) The application must let doctors add and remove exercises to a rehabilitation session.
   (g) The application must let doctors see the rehabilitation session set to a specific patient.
   (h) The application must let doctors see the feedback on each session left by a specific patient.

3.1.2 Non-Functional Requirements

In addition to the functional requirements explained above, the project also has some non-functional requirements. These are gathered by the aspect of the system they are related to:

1. Concurrency
   (a) The system must let multiple users use the application at the same time.
2. Portability
   (a) The application must be compatible for the most android devices possible.
   (b) An internet connection will be necessary in order to use all the functionalities of the application.
3. Security
   (a) To be able to use the application, the user must be logged in.
   (b) A session will be open until the user decides to log out.
   (c) Personal data must be encrypted.
4. Usability
   (a) The application must have an intuitive interface easy to use.
   (b) Elements and actions must be as visual as possible.
   (c) The application must be attractive and appealing to users.
5. Efficiency
   (a) Retrieving any kind of information that users might need must be as fast as possible. Leading to no unnecessary waiting times.

6. Privacy

   (a) All personal data retrieved must be treated following the LOPD (*Ley Organica de Protección de Datos*) rules.

3.2 SWOT Analysis

	Internal Origin	External Origin
Harmful	**Weaknesses** • Only being implemented for Android devices. • Being only one person implementing the whole project. • Not having a wide knowledge in every area necessary to develop the application. • First time working in a project as difficult. • Need of an internet connection to use the application.	**Threats** • Dependence on rehabilitation centres. • The fact that an admin has to register each user (both patients and doctors). • Many times patients prefer to have contact with an human rather than having the information they need on a device. • Risk of the idea being stolen by a stronger company.
Helpful	**Strengths** • Having used before some IDEs such as Android Studio or Adobe XD. • Having a bit of knowledge in every area of the project. • The app will be easy to use and users preferences will be taken into a count. • Every step will be validated either with user tests or by contacting people that have wider knowledge on the aspects treated in order to be sure that the decisions are as correct as possible.	**Opportunities** • Most rehabilitation centres don't have a way to manage patients exercises. • Easy distribution through Google Play • As far as researched, no one has ever implemented an idea like this. • Many users have their own Android device. • Almost everyone that has an Smartphone has a data plan and, as a consequence, a working internet connection. • This idea may change the way that a rehabilitation centre works. • This project could be implemented in every rehabilitation centre.

Figure 10: SWOT matrix.

3.3 Information Gathering

To have information about the needs that doctors had and how potential users use their devices a few tasks were conducted. This process is called Ethnography [23] and it is a qualitative method where researchers observe and/or interact with a study's participants in their real-life environment.

Within the field of usability, user-centred design and service design, ethnography is used to support a designer's deeper understanding of the design problem – including the relevant domain, the audience(s), processes, goals and context(s) of use. It is most useful in the early stages of a user-centred design project. This is because ethnography focuses on developing an understanding of the design problem. Individual methods which are available within an ethnographic study include participant observation, interviews and surveys. All of these ethnographic methods can be very valuable in gaining a deeper understanding of a design problem.

3.3.1 Surveys

In order to obtain information about the knowledge and the kind of devices that people are used to interacting with a survey was conducted. Since anybody could be a patient in a rehabilitation centre, this survey had no restrictions related to the people that could participate
In addition, another survey was created that asked all the questions as the first one but it had some extra questions. Its purpose was to be answered by people that were or had been in a rehabilitation process and the additional questions were related to their experience throughout that process. A brief explanation of the functionalities that the application would provide was included in order to be evaluated and, if they felt like something was missing, a text field was added so that they could add any proposal that might come to their mind.
A total of 88 subjects answered the general survey while the specific survey created for people that were or had been in a rehabilitation process was answered by 11 patients.
In order to see the range of ages of the subjects that participated in the survey provided. The results showed that the age of the people that answered the survey is very diverse, leading to better results since all age ranges are represented.

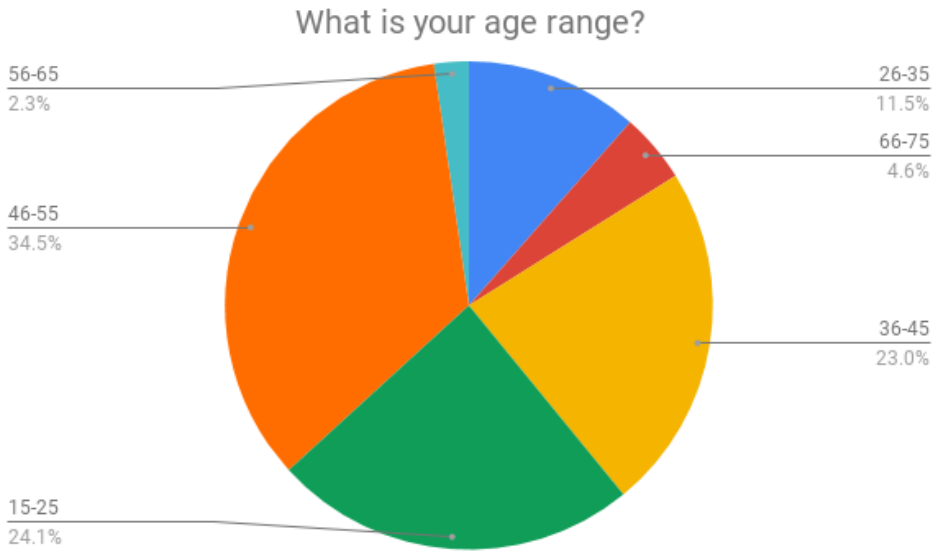

Figure 11: Distribution of the age ranges that participated in the survey.

One of the most important questions was how often potential patients use their devices since this is highly related to how comfortable they feel using them.

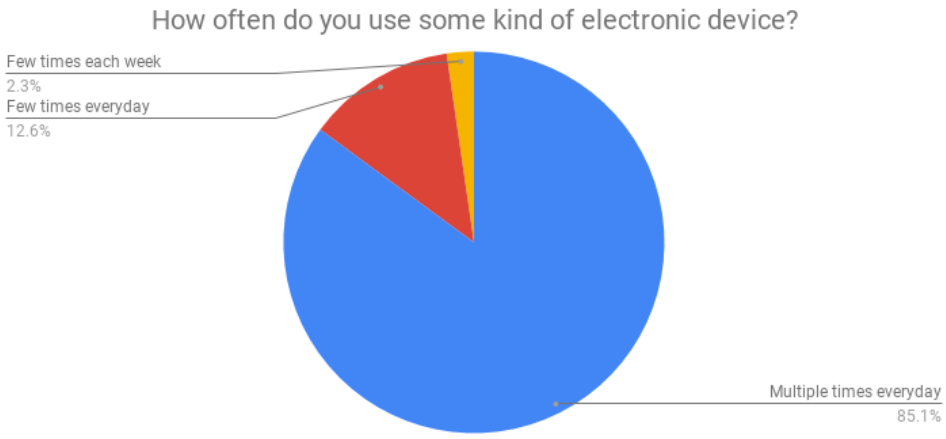

Figure 12: Usage of electronic devices according to the survey.

Once the information about the usage of electronic devices, it was very important to know if they own any other devices apart from computers since this application is targeted to a mobile device. As we can see in the following chart, the results were clear.

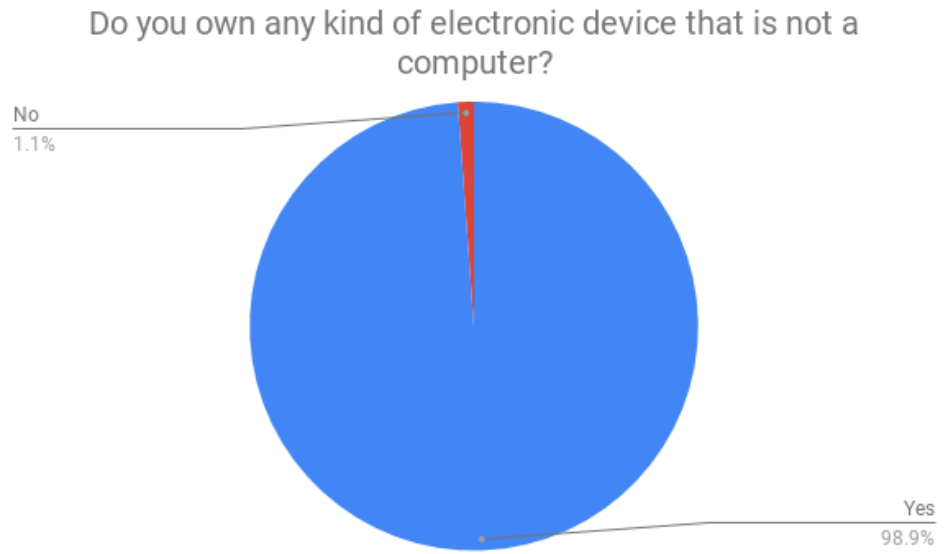

Figure 13: Ownership of devices different to computers according to the survey.

Then, it was really important to know which kind of devices different to computers did they have, since these will be the target of the application.

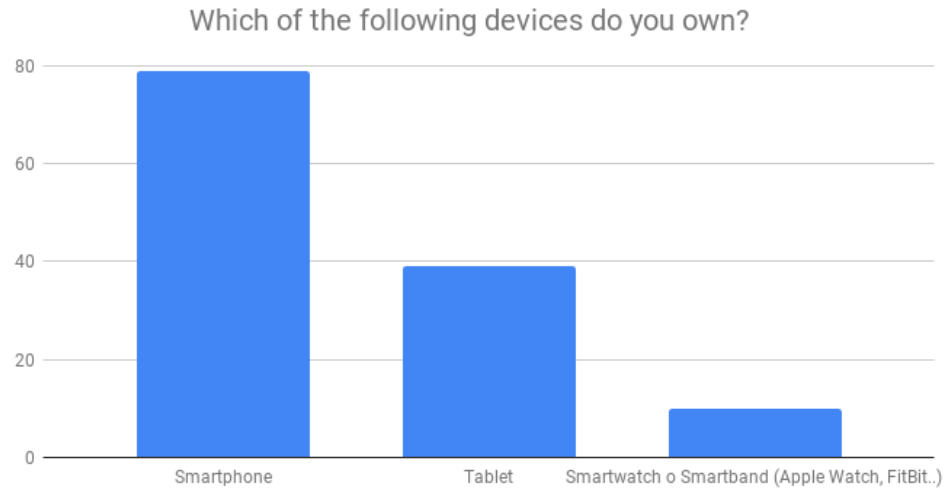

Figure 14: Distribution of devices owned by the subjects that participated in the survey.

As seen, almost all of them have a Smartphone and in addition, a lot also have a tablet. In addition, some of them have some kind of tracking devices such as an Apple Watch or a *Smartband*. As a result, this application should be built mainly for smartphones although it would be really positive to add support for tablets.
Then, it was crucial to know the operating system that runs on the devices owned by the subject in order to build the application for as much of them as possible.

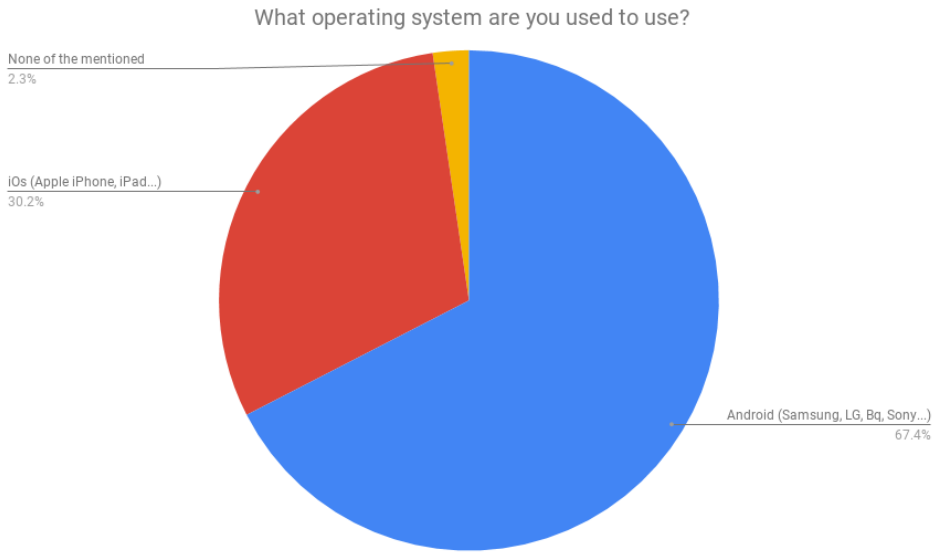

Figure 15: Distribution of the different operating systems according to the survey.

As we can see, the main operating system used by the subjects is Android, followed by iOs and the least common answer was the one representing devices running operating systems less popular such as Windows Phone or Blackberry OS.

This means that developing an application for Android would work for almost 70% of patients approximately. As a consequence, to start this project, developing an A for Android and providing devices for users that might not have one of their own would be enough to see if this idea is successful. Then, if it succeeds, it would be time to build the respective version for iOs.
In addition, it was important to be aware of what they use their devices for, since that would give us information about how they use their devices and if they do many different tasks with them.

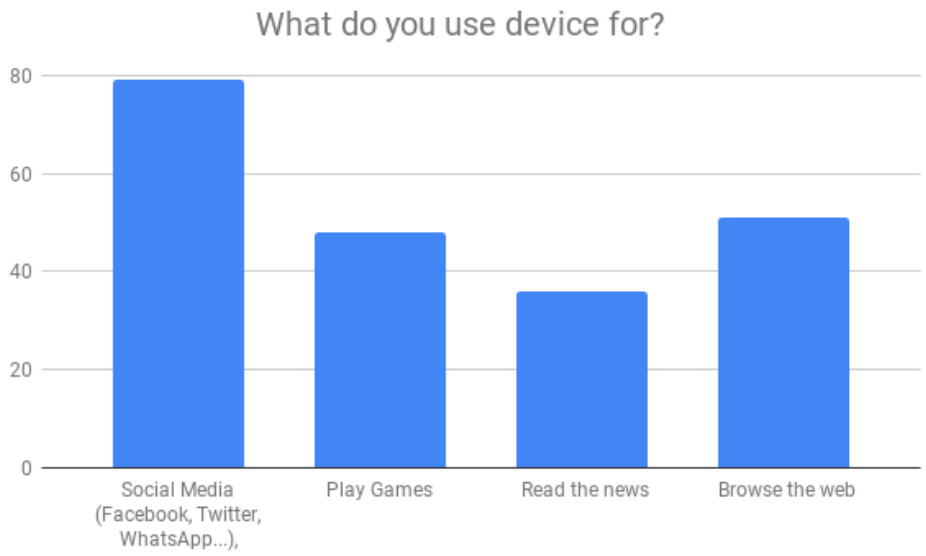

Figure 16: What subjects use their devices for according to the survey.

As we can see, almost everyone uses their devices to do more than one thing. Nevertheless, most of them use it mainly for social media applications that have a structure and a way to use them very similar to the one that will be implemented for this project.
As a result, people are very used to interact with their devices and most of them do it numerous times a day and for many different reasons. As a consequence, almost all of them should be comfortable using different applications and the inclusion of the one resulting from this project wouldn't be anything dramatic.
Last but not least, it was very important to acknowledge how many users had some kind of disability that affected in the way they interact with devices, in order to be able to develop the application being aware of them.

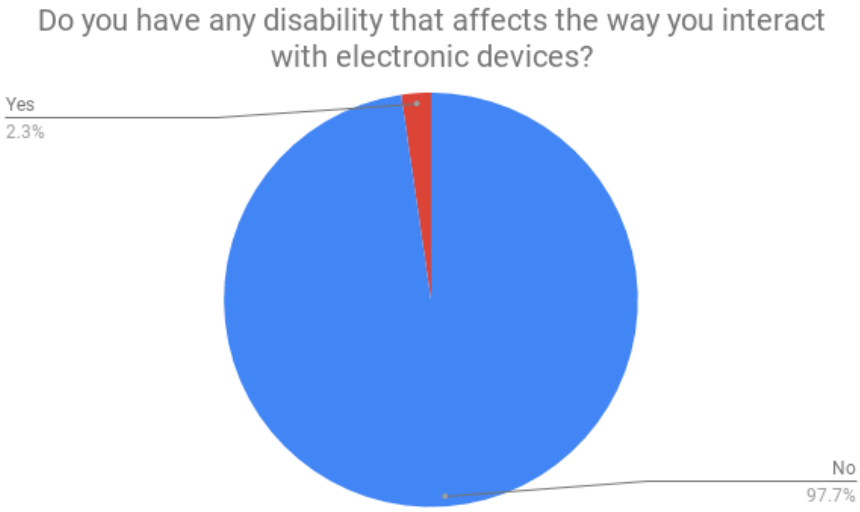

Figure 17: Percentage of users with disabilities that affect the way they interact with electronic devices.

As seen in the chart, a small part of the subjects had a disability that affected the way they use electronic devices. Most specifically, they were two of them and both had a hearing disability. The application that is being developed should not have any functionality that implies having to hear anything so, at first, these potential patients shouldn't have any problems using it. Nevertheless, the application must be designed and built to be as accessible as possible in order to be usable by people that may have other disabilities and have not been able to participate in the survey.
As mentioned before, there was another version of the survey created for people that had been through a rehabilitation process in order to see their opinion about the application. The survey was exactly the same as the one for all people but with two additional questions.
At the beginning of the survey, the application and its functionalities were explained so that patients could understand the capabilities of the system and give their opinion. Then, they were asked if they would find an application like the one described useful and the results were really great.

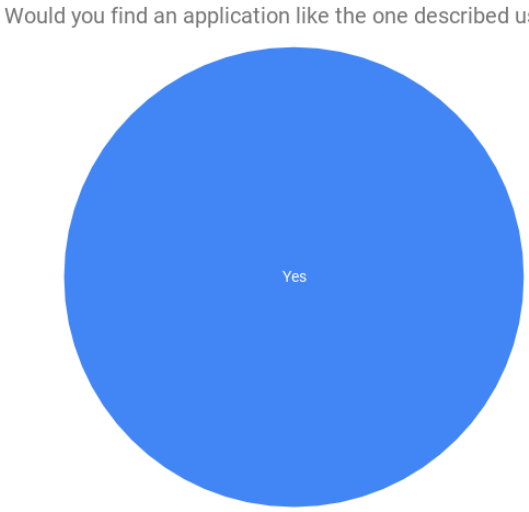

Figure 18: Patients that think that the application would be useful

As we can see, all of the patients that took the survey thought that an application like the one described in this project would be really useful. In addition, they were asked if they needed any functionality that was not included in the initial description. Only one of the subjects proposed the functionality of being able to be able to get notifications when the time of an appointment is approaching.

3.3.2 Meetings with doctors

A series of meetings with doctors were conducted in order to acquire some knowledge about technical terms that could be necessary to understand a rehabilitation process and to see their point of view on the application.
All the interviews were conducted in the same way, at first, some questions were made in order to get the profiles of users that they often see in their centres.
The result was that as expected, that they don't have a specific profile of patients since all range of ages are vulnerable to injuries and conditions that might lead them to a rehabilitation process. Nevertheless, older people are more prone to have an injury although their rehabilitation sessions might be different.
In addition, people with disabilities are not treated as the other patients, since they have someone of the medical crew with them throughout their session, so this project would not be useful to them. As explained by many doctors, this application would mainly help people with traumatic injuries, since these are the ones that can be treated with specific physical exercises.
Once the profiles and the additional information about rehabilitation processes were obtained, the application and its functionalities were explained to the doctor. Then they were asked if they thought it was useful and if they had any additional functionalities that the application could include in order to improve the effectiveness of their work.
All of them agreed that the application would be very useful. In addition, most of them thought that the functionalities were enough but one doctor that suggested that the application should let patients do their exercises at home since there are some rehabilitation processes that don't require any kind of specific tools. This would lead to a less crowded centre and as a result, into doctors being able to do their job more comfortably.

4 Design

Once the requirements were established and basic knowledge of the area that will be worked on was achieved it was time to start designing the application.
This process is basically the creation of a prototype of the application both for the patients and the medical crew using the knowledge acquired in previous phases of this project and taking into account every requirement and functionality established.
This prototype must simulate both the design of the future application and the functionalities that it will have since it is crucial to carry out some tests with users to spot the weak points of the design and the way the system works and fix them in the process of creating the final application. This is because the earliest mistake is spotted, the easiest it is to fix. As a consequence, making a prototype as close to the application as possible will lead to finding the most mistakes possible.

4.1 Tools used

To build the prototypes both for the patients and the medical crew a software named Adobe XD was used.
Adobe XD [24] is a vector-based tool developed and published by Adobe Inc for designing and prototyping user experience for web and mobile applications. The software is available for macOS, Windows, iOS and Android. XD supports vector design and website wire-framing, and creating simple interactive click-through prototypes.
One of the strongest points of this tool is the fact that is really easy to get used to the way it works since it works the same way as all the other tools created by Adobe such as Photoshop.

It also has cloud integration and makes it really seamless to work with different devices and to share the different versions of the prototype.
In addition, it has a mobile application that makes really easy to transfer this prototype to a real device and make it look like it is a functional application. This is really important when it comes to testing the prototypes since it leads to a simulation of the interaction with the application almost as if it really was the final version. This is because it the prototype simulates the functionalities that the application will have and the fact that is running on a device creates an environment were the testers feel comfortable and interact the same way as they would with the real application.

4.2 Main aspects

First of all, it was very important to give the application identity in order to be easily identified. This identity is basically created by the colours used in the design and the logo that will represent it.
Although it can look like it is easy to choose the set of colours to be used, it was very hard since it had to be an appealing design and many aspects such as the accessibility had to be taken into account. As a consequence, many articles about how to choose a colour for the design of an application in order to create a great user experience were read to acquire some knowledge about the subject. The main points[25] were the following.
There is a significant difference between the preference between genders when coming to colour selection. The study was done by Joe Hallock to find out about most favourable and least favourable colours and there was significant favour for the blue colour of both men and women and the orange colour was the most disliked colour by both men and women. In this study, it was found that men preferred Bold colours and women preferred soft colours. The distribution of preferences can be seen in the following figures.

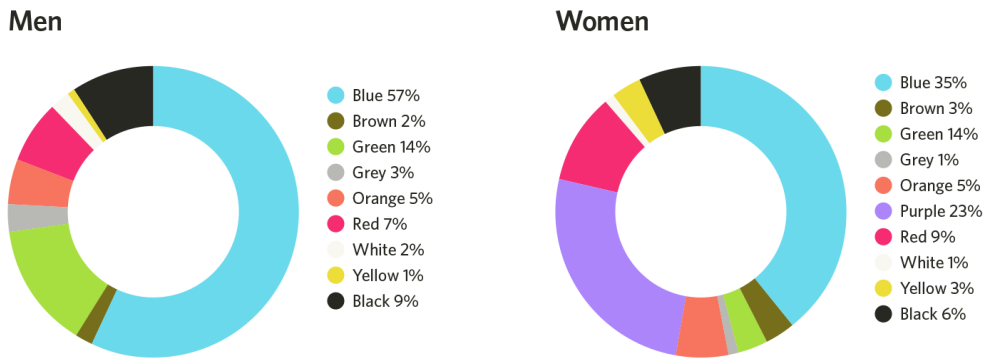

Figure 19: Most liked colours by men and women according to Joe Hallock's study [26].

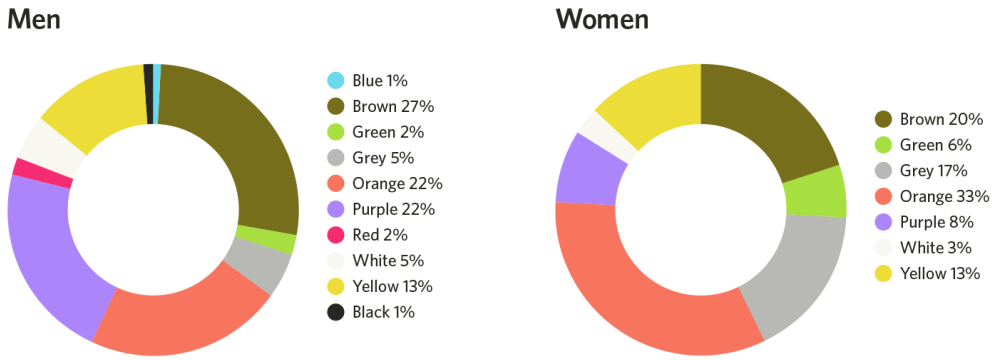

Figure 20: Most disliked colours by men and women according to Joe Hallock's study [27].

The product design is not just about being understandable but also discoverable. Our brains always like to focus on brands that are immediately recognisable. In order to create the product look engaging and recognisable, colours must be used properly and be aligned with business ideas, personality, emotion and differ from the competitors. Many studies have shown that colour is a key fact when come to deal with direct competitors.
Each colour has also a set of characteristics that relates to them. For example, blue is considered to give emotions such as trust, safety and relaxation. The light blue colour creates emotions such as calm and makes the user feels refreshed. The blue colour is also associated with happiness. Normally the clear blue sky gives that feeling of happiness and friendliness towards the user. By using the fact of being friendliness the trust is created towards the user.
This describes almost perfectly what our application must transmit to users and, in addition, blue is also the most liked colour by everyone. As a consequence, blue was chosen as the main colour for the project.
Then, the logo of the application had to be designed. It is really important to create an appealing logo since it is the first thing that future users will see and it will represent the whole project.
A crucial aspect came across when thinking about the logo of the application and it was the naming of the application. It had to be something short and catchy so users find it easy to remember. Many names were thought including Rehabilit-App or Rehabiliapp but they were very long and complex to pronounce depending on the language. At last, the name chosen was Rehapp, since it contains both the concepts of rehabilitation and application. In addition, if it's said out loud, it sounds like rehab which is the short term for rehabilitation.
After many designs, the chosen one was designed using Photoshop and it was the name of the application written in white with a blue background. The blue shade in the background is also the same shade used consistently throughout the application design. In addition, it will also be the icon of the application.

Figure 21: Logo of the project.

Once these pillars of the design of the application were established, it was time to start designing the prototypes.

4.3 Prototype

Two versions of the prototype had to be created since although it is the same application, patients and medical crew will have two different versions of it. This is due to the cause that each group needs its own functionalities. Nevertheless, there are a lot of things that will be shared between them such as the design or the user experience.
In both cases, the navigation throughout the application would use buttons in order to use its functionalities. It was designed to have less screen depth as possible so users can go to the main screen in the shortest time possible if needed.

Using the flat user interface and an intuitive navigation pattern should lead into a very easy application to use and making both versions as similar as possible would lead into very similar user experience and finding errors or weak points in any of them would lead into improving them both.

4.3.1 Version for patients

Once all the functionalities were listed and the main design aspects were decided, it was time to design the prototype. There were mainly 8 screens with some variations to design and the workspace ended up looking like the figure attached below.

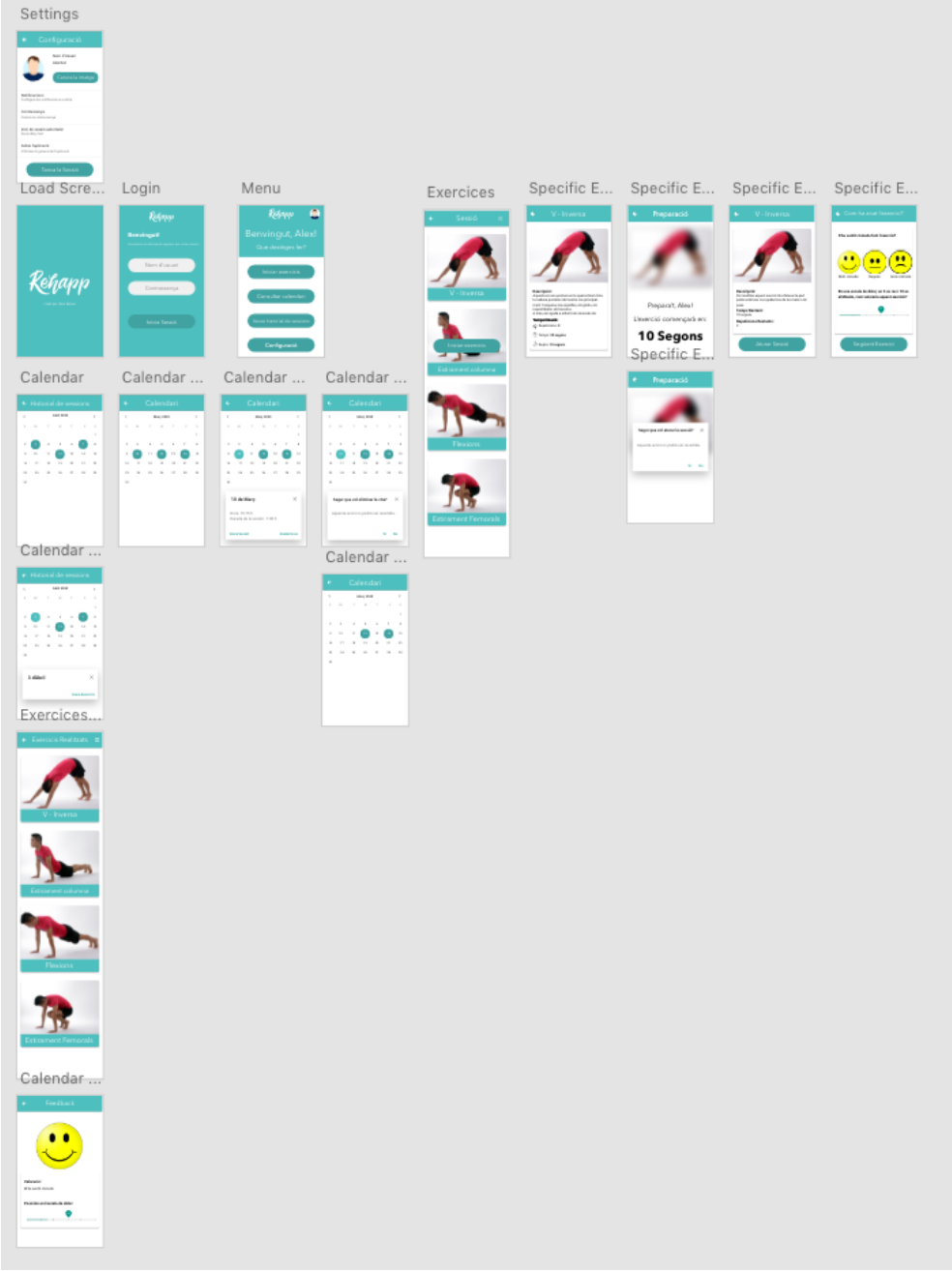

Figure 22: Patient's prototype workspace

Splash Screen
This screen should provide information about the application and give some identity to it. It will appear only when the application is yet to be started.

Figure 23: Splash screen.

Login
With this screen, users will be able to introduce their credentials in order to log into the application and be able to use its functionalities.

Figure 24: Login screen.

Main Screen
This will be the main screen representing the main functionalities of the application. As we can see, we have three buttons representing them and an extra one that would let users establish their settings.

Figure 25: Patient's Main Screen.

Settings
This screen will let users establish their preferences about their personal information and the ones related to the application settings.

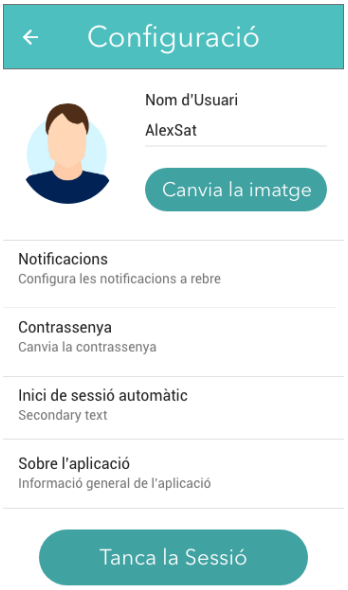

Figure 26: Settings screen.

Exercises List Screen
This screen will show the exercises that the user has to do on a specific session. In addition, if an exercise is clicked, the information about it will be shown.

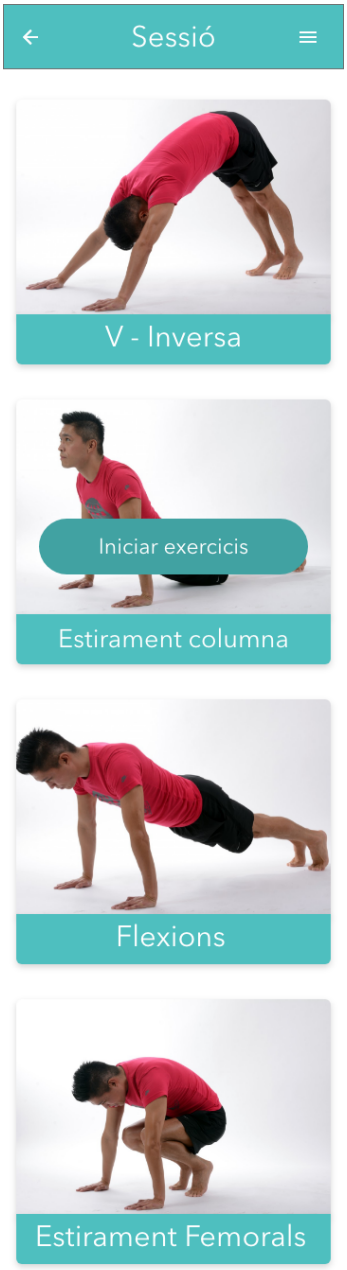

Figure 27: Screen showing the list of exercises available.

Specific information about an exercise
This screen will show the information about an exercise such as its description or the main aspects related to it.

Figure 28: Screen showing the information about an specific exercise.

Calendar
This screen is used both in the functionality of seeing future appointments and the one that shows the information about the sessions that are already finished.

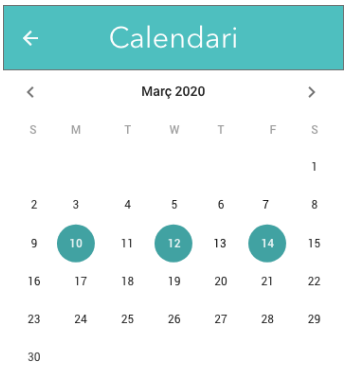

Figure 29: Calendar screen.

As a consequence, many screens related to the functionalities mentioned had to be designed. Attached below, we can see the screens related to a future appointment.

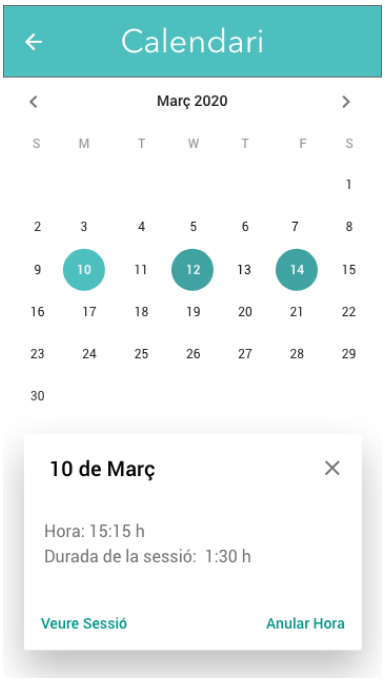

Figure 30: Information about a future appointment.

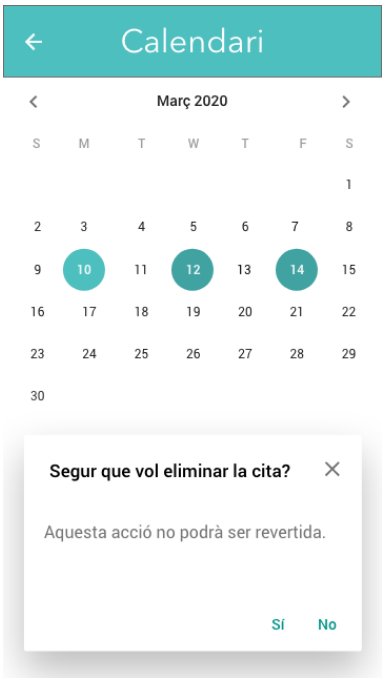

Figure 31: Appointment cancellation screen.

The screens related to checking the information of a session already finished would be the following.

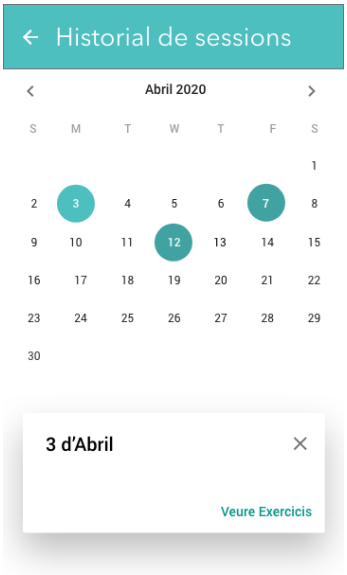

Figure 32: Screen showing the pop-up to see the exercises done at a specific session.

As seen, this pop up would lead into the exercises screen seen before with the information related to the specific session.

Exercises to do throughout a session.
These screens would let patients know the timings and the exercises they should do in order to carry out the session properly.

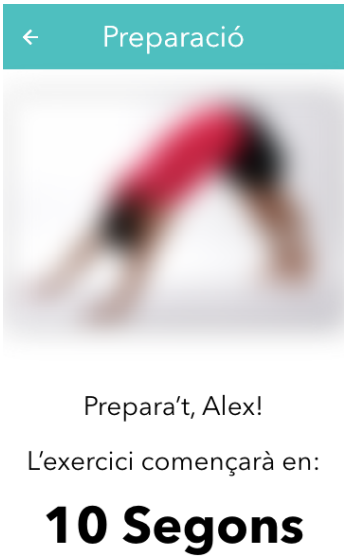

Figure 33: Screen showing the countdown to start an exercise.

The figure attached above shows the main screen a user would see when they start a session, leaving them a few seconds to get ready to start the exercise. Once they are ready, a screen like the one seen below will show the main information about an exercise and the timings.

Figure 34: Screen seen while doing an exercise.

When the exercise is done, a screen like the one attached below will let patients leave feedback on the exercise done in order to track their progress.

Figure 35: Screen to let patients leave feedback on an exercise.

Patients can also stop a session if anything occurs. A pop up would let them confirm they want to do so in order to avoid stopping a session involuntarily.

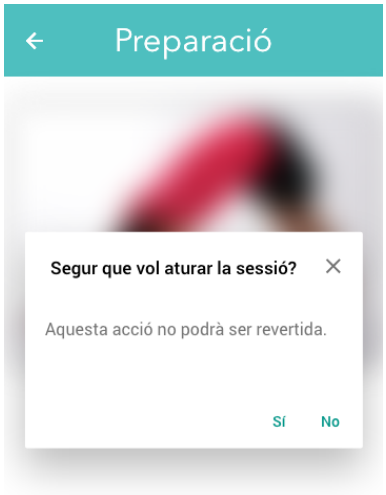

Figure 36: Pop-up to confirm that an patient wants to stop a session.

4.3.2 Version for doctors

This version would share all the screens that represent the shared functionalities. Nevertheless, there were many screens that were yet to be designed in order to represent the specific tasks that the medical crew would do.
The workspace associated with the doctor's prototype ended up looking like it is shown on the figure attached below.

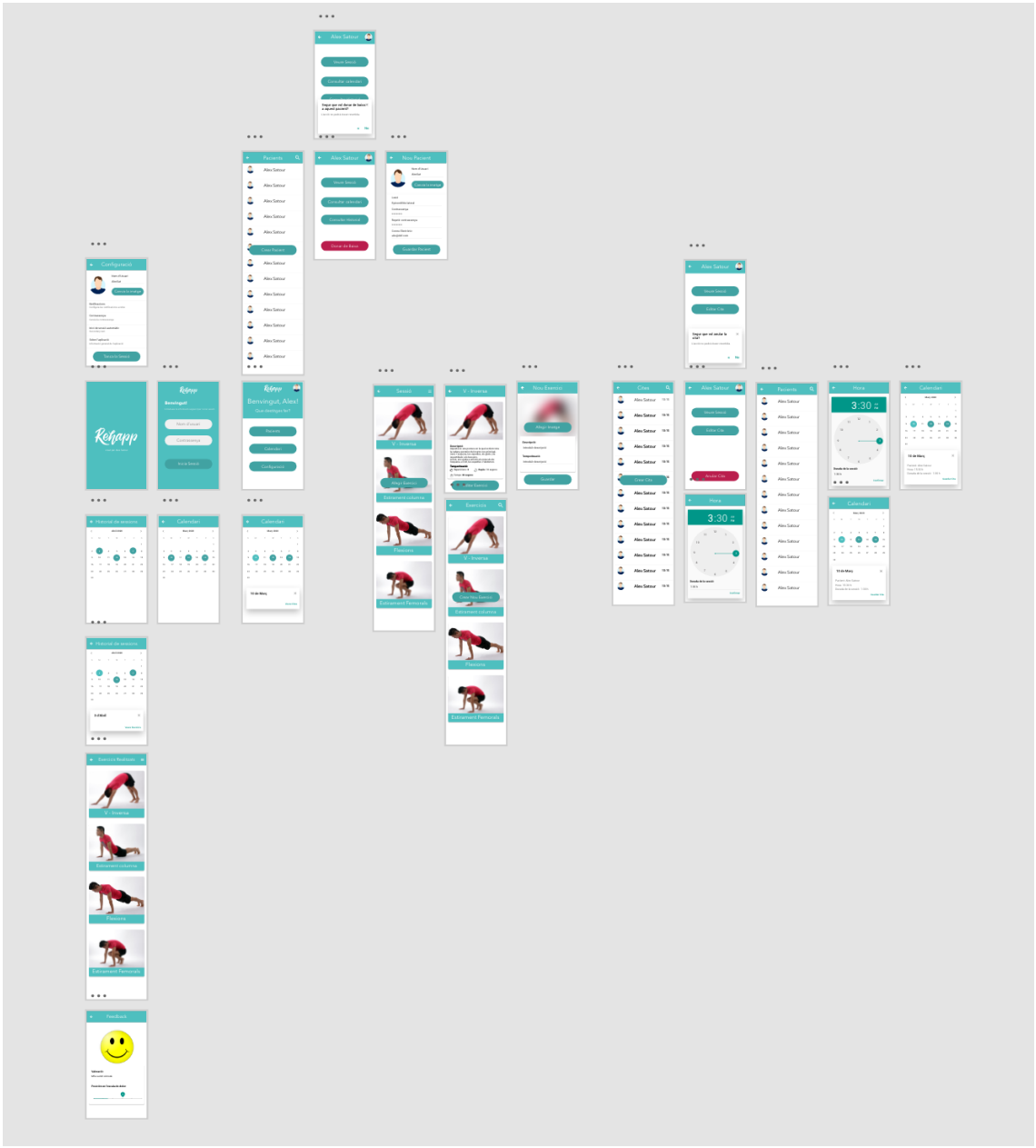

Figure 37: Doctor's prototype workspace

Main Screen
As seen with patients, this will be the main screen that will let doctors use the functionalities provided by the application.

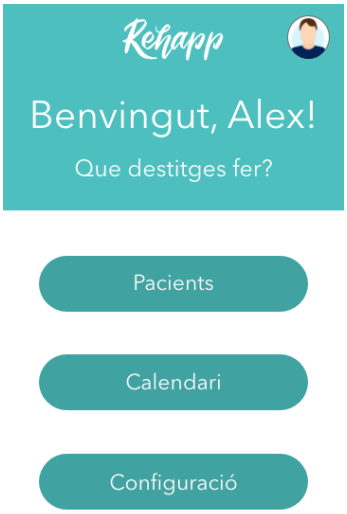

Figure 38: Doctor's Main Screen.

Patients Management
When the patient's button is clicked, a screen like the one attached below will be shown. It will let doctors see the patients that are currently on a rehabilitation process.

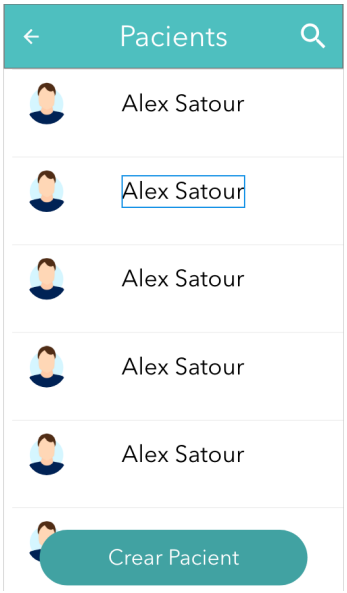

Figure 39: Screen showing the list of patients.

If a doctor selects a patient, the following screen will be shown.

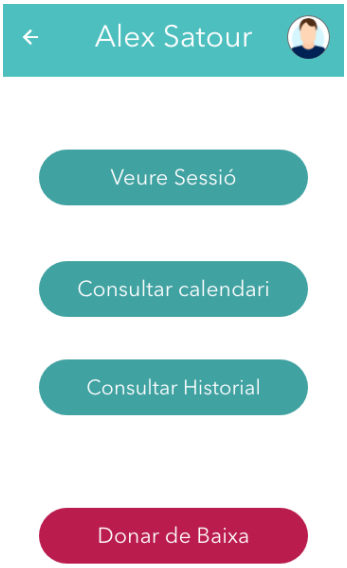

Figure 40: Specific actions that can be done on a patient's user.

As we can see, a doctor would be able to see and edit the session of a patient, check its future appointments or the past sessions. In addition, they could also delete their profile if their rehabilitation process has ended. To avoid possible mistakes, a pop up will be shown in order to confirm the action of deleting a profile that belongs to a patient.

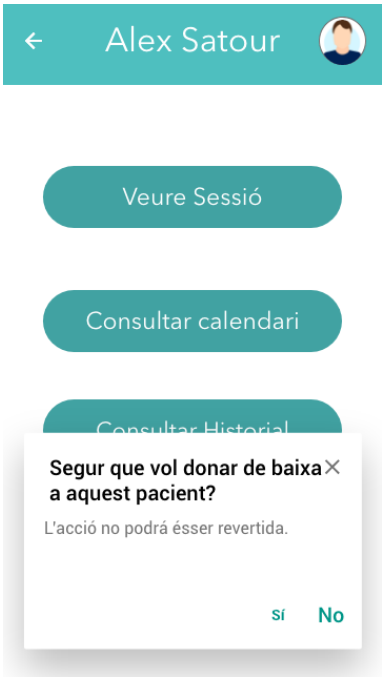

Figure 41: Pop up shown to confirm the action of removing a patient's profile.

If the doctor wants to create a profile for a new patient, they would simply have to click the button seen on the patient's list and the screen attached below would appear in order to fill the details and create the profile.

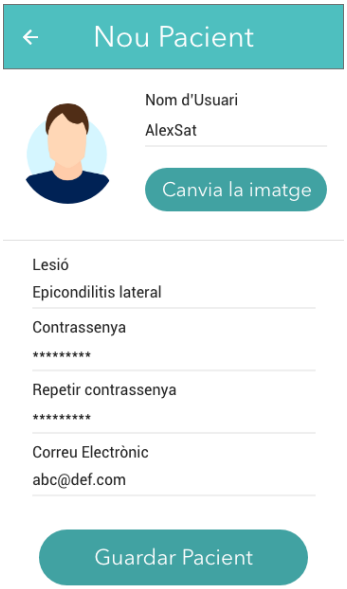

Figure 42: Screen to create a profile for a new patient.

Exercises Management
Doctors must be able to see the session of a patient and add exercises to it if necessary. The screen that should let doctors do this action will be like the one attached below.

Figure 43: List of exercises in a session.

In order to be able to easily add an exercise to a session, a screen that lists all the available exercises was designed.

Figure 44: List of exercises available.

When a doctor clicks the button associated with the creation of a new exercise, the screen attached below would appear so they can create it.

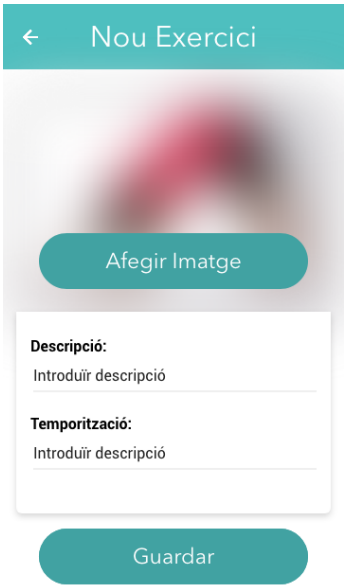

Figure 45: Screen shown to create an exercise.

If an exercise is selected, a screen showing up its details would be shown.

Figure 46: Specific Information of an exercise.

If a doctor wants to edit an exercise, a screen like the one seen when an exercise had to be created would appear to let the doctor modify the necessary aspects.

Appointments Management
Doctors will have a screen that will contain a calendar like the one shown on the patient's versions. When they select a specific day, a screen like the one attached below will show the appointments on that specific date.

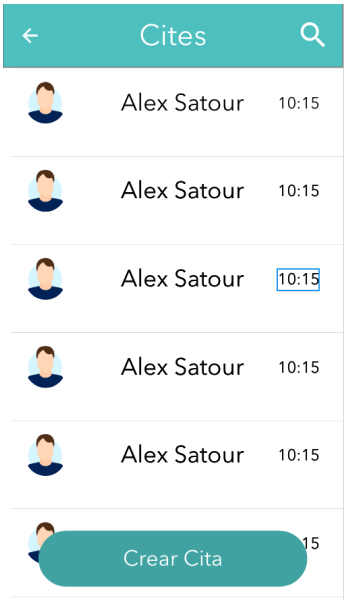

Figure 47: Appointments on a specific day.

If a doctor selects the button associated with the creation of a new appointment, a screen would appear to let them set the details about it.
Firstly, a screen like the one below will let doctors choose the length of the appointment.

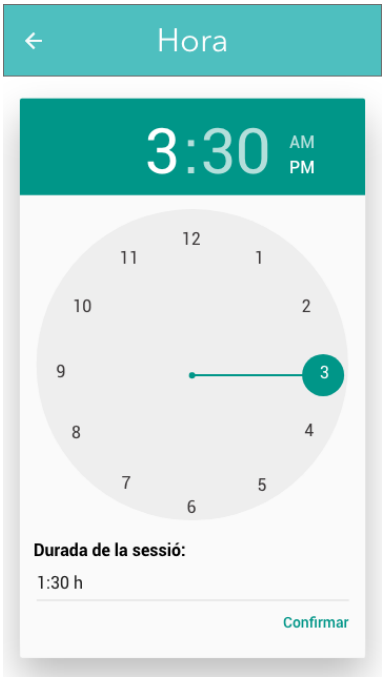

Figure 48: Screen to set the length of an appointment.

Then doctors should choose from the list of patients the person associated with the appointment.

Figure 49: List of patients.

Once these aspects are selected, a pop up will show the main aspects of the appointment so that doctors can confirm them and create the appointment.

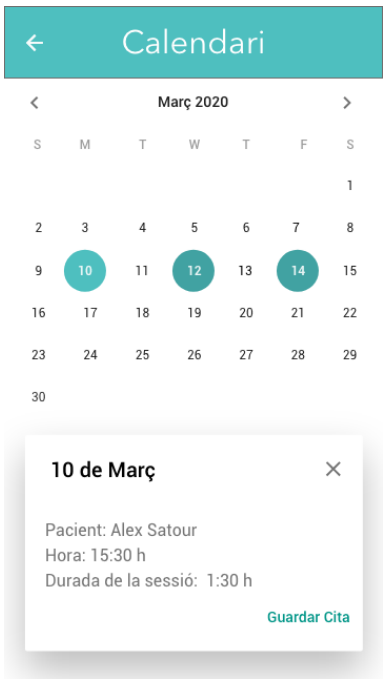

Figure 50: Pop-up showing the details of the appointment about to be created.

Doctors could also select an appointment of the list in order to see its main aspects, edit or delete it.

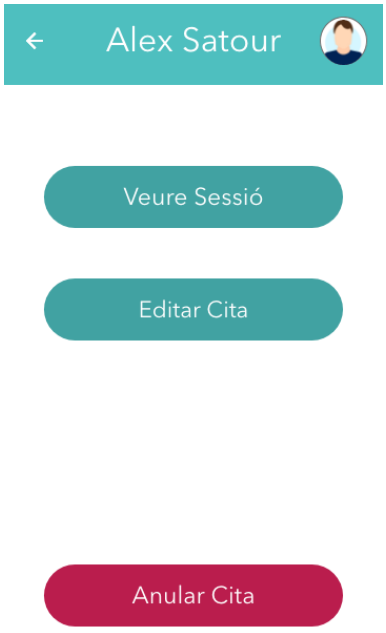

Figure 51: Screen showing the available actions on a scheduled appointment.

If a doctor chooses to edit the appointment, the screens that will appear would be like the ones shown on the creation of an appointment letting them edit each aspect.
On the other hand, if the doctor wants to cancel it, a pop up will appear in order to avoid any unwanted actions.

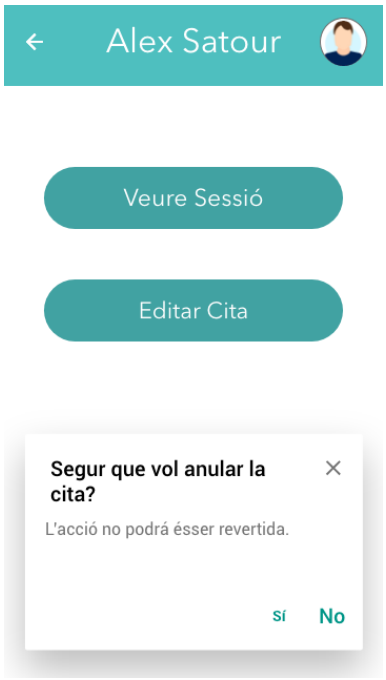

Figure 52: Pop-up shown to avoid an unwanted deletion of an appointment.

4.4 Tests with users

Tests with users were carried out in order to spot the mistakes done whilst creating the application. This process is really important because as a developer is really difficult to be able to spot weak points in your own design. In addition, there is only one person working on this project and most of the decisions have been taken under only one point of view.
As said before, these tests also reduce the costs of fixing mistakes on a project, since the sooner they are spotted, the easier and cheaper it is to fix them.

4.4.1 Setup used

In order to recreate the environment that users would have on real usage of the application, a real smartphone was used to perform these tests. It was an Android device that had the Adobe XD application installed in order to be able to run the prototype on it and make the experience as realistic as possible.
The tests were carried out in spaces where the application would be used, so all external factors were also simulated in order to test all the possible aspects of the applications, such as if the text is readable in the environment in which the application is used.

4.4.2 Subjects

Participant	Gender	Age	User/Medical Crew	Uses applications regularly
1	Female	14	User	Yes
2	Male	17	User	Yes
3	Male	17	User	Yes
4	Male	21	User	Yes
5	Male	22	User	Yes
6	Male	22	User	Yes
7	Male	22	User	Yes
8	Female	28	User	Yes
9	Female	34	User	Yes
10	Female	36	User	Yes
11	Male	37	Doctor	Yes
12	Female	39	Doctor	Yes
13	Male	40	User	Yes
14	Male	42	User	Yes
15	Female	47	User	Yes
16	Male	52	User	No
17	Female	56	User	Yes
18	Female	68	User	No
19	Male	69	User	Yes

Table 1: Information about users.

In the end, 19 users tested the prototype. As we can see, on the user's side all age ranges were represented by at least one tester, which led into a better testing process.
Due to scheduling issues, the doctors that collaborated throughout the project were not able to participate in the testing process.
Two doctors tested the prototype and since they were the only ones that were going to do it, these tests were carried out in a very detailed way trying to spot all the improvements that they thought could be done.
Nevertheless, both versions are very similar and most of the issues spotted on one must be fixed in the other. As a consequence, the fact that it was not possible to test the doctor's prototype as expected did not affect as much as if the versions were completely different.

4.4.3 Testing process

First of all, testers were given a brief description of the project so that they can understand the project that is being developed. Then, they were reminded that the application is what is being tested and not them, so if they felt uncomfortable they could quit the test at any moment. It was very important to clarify that no questions could be answered in order to be as objective as possible.// The test consisted of performing a set of tasks that final users would do on a regular basis measuring the efficiency, the usability and the satisfaction level. Efficiency is measured dividing the numbers of steps done by the user by the steps on the optimal path. On the other hand, satisfaction will be measured asking the subjects the level of difficulty of each task and performing a System Usability Scale test.
The System Usability Scale[28] (SUS) provides a reliable tool for measuring usability. It consists of a 10 item questionnaire with five response options for respondents; from Strongly agree to Strongly disagree.
Interpreting scoring can be complex. The participant's scores for each question are converted to a new number, added together and then multiplied by 2.5 to convert the original scores of 0-40 to 0-100. Though the scores are 0-100, these are not percentages and should be considered only in terms of their percentile ranking.

Based on research, a SUS score above a 68 would be considered above average and anything below 68 is below average, however, the best way to interpret results involves "normalising" the scores to produce a percentile ranking.

Patients tasks

1. **Log-in and log out**
   Description: You want to log-in in our system since is your first time here and then you want to log out since you are done using the application.
   Objective: It is an easy task which should not lead to any issue. The main purpose of the task is to familiarise the tester with the application.
   Optimum path: Three steps:
   (a) Log-in
   (b) Configuration
   (c) Log out
2. **Check the information about an exercise**
   Description: A new exercise has been added to your session and you need to know how to do it properly.
   Objective: Test the functionality and prevent possible issues.
   Optimum path: Two steps:
   (a) Start exercises
   (b) Choose exercise
3. **Start a rehabilitation session and then cancel it**
   Description: You have arrived to your session and you have to start the exercises but all of a sudden, you have to stop doing them.
   Objective: Test the functionality and prevent possible issues.
   Optimum path: Four steps:
   (a) Start exercises
   (b) Start session
   (c) Stop session
   (d) Accept pop-up
4. **Cancel an appointment**
   Description: You have to cancel the appointment on the 10th of March.
   Objective: Test the functionality and prevent possible issues.
   Optimum path: Four steps:
   (a) Calendar
   (b) Choose day
   (c) Cancel appointment
   (d) Accept pop-up
5. **See the exercises of a future session**
   Description: You want to see the exercises you will have to do on the 10th of March.
   Objective: Test the functionality and prevent possible issues.
   Optimum path: Three steps:
   (a) Calendar

(b) Choose day

(c) Check exercises

6. **Check the feedback left on an exercise**
   Description: You want to check the feedback left on an exercise of the session done the 3rd of April.
   Objective: Test the functionality and prevent possible issues.
   Optimum path: Four steps:

   (a) Check session history

   (b) Choose day

   (c) Check exercises

   (d) Choose exercise

Medical crew tasks

1. **Log-in and log out**
   Description: You want to log-in in our system since is your first time here and then you want to log out since you are done using the application.
   Objective: It is an easy task which should not lead to any issue. The main purpose of the task is to familiarise the tester with the application.
   Optimum path: Three steps:

   (a) Log-in

   (b) Configuration

   (c) Log out

2. **Create new patient**
   Description: A new patient has arrived and its profile has to be created.
   Objective: Test the functionality and prevent possible issues.
   Optimum path: Three steps:

   (a) Patients

   (b) Create new patient

   (c) Save the information

3. **Check the session saved for a patient and add a new exercise**
   Description: You want to add a new exercises to a session.
   Objective: Test the functionality and prevent possible issues.
   Optimum path: Five steps:

   (a) Patients

   (b) Select the patient

   (c) Check session

   (d) Add exercise

   (e) Choose the new exercise

4. **Check the feedback that a certain patient left on an exercise**
   Description: You want to check the feedback that a certain patient left on an exercise of the session done the 3rd of April.

Objective: Test the functionality and prevent possible issues.
Optimum path: Six steps:

(a) Patients

(b) Select the patient

(c) Check History

(d) Choose day

(e) Check exercises

(f) Choose exercise

5. **Check the appointments and delete one of them**
Description: You want to delete an appointment set for the 10th of March.
Objective: Test the functionality and prevent possible issues.
Optimum path: Five steps:

   (a) Calendar

   (b) Select date

   (c) Check appointments

   (d) Choose the one to be deleted

   (e) Delete it

6. **Create a new appointment**
Description: You want to create a new appointment for a patient named Alex Satour and save it.
Objective: Test the functionality and prevent possible issues.
Optimum path: Seven steps:

   (a) Calendar

   (b) Select date

   (c) Check appointments

   (d) Add a new one

   (e) Select time

   (f) Select Patient

   (g) Save it

4.4.4 Results

Once the testing process was finished it was time to evaluate the results. As said before, there were mainly three aspects to be evaluated: the efficiency, the effectiveness and the satisfaction.

Efficiency
To be able to measure the efficiency of the application we counted the number of steps each user did to perform each task. The optimal path was also calculated in order to have a reference to compare them to. The results associated with the patient's prototype can be seen in the table attached to the next page.

Task	Path user	Shortest Path
1	3.2	3
2	2.3	2
3	4.5	4
4	5.1	5
5	3.2	3
6	4.6	4

Table 2: Average and optimal number of steps for each task of the patients prototype testing process.

As we can see in the table above, the results are good but not optimal. While users were performing the tests, users interaction with the prototype was studied in order to identify the reasons behind the issues. In this case, most of them were related to bad naming that led into confusion.
In the other hand, the results of the doctor's prototype testing were the following:

Task	Path user	Shortest Path
1	3	3
2	3	3
3	5	5
4	8	6
5	5	5
6	7	7

Table 3: Average and optimal number of steps for each task of the medical crew members prototype testing process.

As we can see, all the tasks were done in the optimal steps. Nevertheless, one of them wasn't able to finish a task. Leaving that apart, the results are very good but it has to be taken into account that only two users tested the application. As explained before, both doctors were asked to express their ideas about ways of improving the application in order to fix the errors that could not be spotted in these few tests.

Effectiveness
All users testing the patient's prototype were able to finish all tasks given. This is really important because it means that although some UI elements were confusing as seen in the efficiency section, due to the overall structure of the application users were able to complete the tasks. In the other hand, an user testing the medical crew members prototype wasn't able to finish the task. It was due to a confusing name of a button because it was named Calendar and the doctor thought that in that section he would also be able to see the past sessions of a patient.

Satisfaction
As said before, satisfaction was evaluated following the idea of the SUS questionnaire. They had a total of 10 statements they had to answer to in a scale from 1 to 5, being 1 that they strongly disagree with the statement and 5 that they completely agree with it. Positive and negative statements were alternated to be able to spot if someone simply chooses random answers.
To be able to obtain a value of satisfaction, we left the value of a certain statement as it was if it was positive, but obtained the inverse value if it was negative. The final results were the following.

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Final
4.9	1.4	4.3	1.1	4.6	1.0	4.7	1.0	4.4	1.0	95.5

Table 4: Average score of each SUS question and final mark.

At the table above, we can see the average score obtained on each question as well as the final satisfaction result. In the end, the average of 95.5 % . This is a very good mark and shows that the design of the application is good.

Improvements
Although the results of the aspects evaluated were really good, there were some points that could be improved. These were either spotted when seeing users interacting with the application or when users themselves manifested some kind of improvement since they were told to express anything that came to their minds while they were doing the testing process.
Nevertheless, one of the strongest points seen throughout the process is that the application barely needs any data input using the keyboard since almost everything is selected via buttons or gestures. This led to guarantee that the data introduced will have no errors and that the interaction with the application is very easy.
With that being said, the issues that could be improved are the following:

1. **Patients prototype issues:**
   (a) **Session button naming:** Some users were not able to do the second task in the optimal steps because the button only specified session and they did not think that exercises could be found in there too. This button should be renamed to Exercises and Session.
2. **Doctors prototype issues:**
   (a) **Calendar button naming:** This button should be renamed to Future Appointments since its name led into confusion.
   (b) **Manage exercises screen:** Both doctors that participated in the testing process suggested an extra button on the main screen that would let them manage the exercises and see the specific information of each case. This would let them show an exercise to a patient much easier without having to go to their profile.
   (c) **Simplify the process of creating an appointment:** Both doctors thought that the process of setting up an appointment took to many screens. This could be simplified into one simple screen where a doctor would add progressively the details asked using a popup to choose the date and an additional screen to choose the patient. This would let doctors know the information they set at any given moment and fix it if necessary, not having to wait until the end as they had to do in the prototype.
3. **Common issues:**
   (a) **Too many screens when seeing the appointments of a day:** In the screens that contained a calendar, space below was wasted since the appointments or the sessions in that day were shown in a separate screen. Having that much space below should be used to show the events in there, removing one screen and making the navigation scheme much simpler.
   (b) **Change screens for pop-ups:** When designing the prototype, many actions such as the one related to a specific appointment were shown in a screen. This created an unnecessary complexity since they could be inside a pop-up screen. Making the whole process much easier.
   (c) **Remove splash screen when unnecessary:** Splash screen should only be shown when the application is loading resources and still is not able to start at a given moment. If the application is ready, the splash screen should not be shown since it will remove unnecessary waiting times and lead into a better user experience.
   (d) **Feedback screen**: Showing the feedback could be simplified showing averages on the event of each session. And, if needed, when a specific session is clicked, a simplified list could be shown, letting know only the necessary information such as the name of each specific exercise and the feedback information that a patient left.

(e) **Show a progress dialog when loading information:** Although this functionality could not be represented in the prototypes, it is really important to let users know what the application is doing at any given moment. As a consequence, since the application relies on downloading many resources from the internet, a screen letting the user know that information is being retrieved should be shown when the application is downloading any kind of content.

(f) **Sign out button:** Due to design language reasons and to improve the intuitiveness of the application, the sign out button should be red in order to let users know that the action is not reversible and they will have to sign in again.

(g) **Project credits moved to a separate section:** Instead of showing the credits of the project at the splash screen, an about section should be created at the setting screen in order to keep the branding of the application and the information about it separated.

As a consequence, fixing these weak points when implementing the real application will lead to its best version possible.

5 Development

In this section the main points about developing the project and implementing all the functionalities needed will be described. Of course, taking into account everything learnt in the phases done before.
All the source code of the final application project can be found in the Github repository[29] associated to the project. Nevertheless, to have it working the files providing the information about google services must be added in order to connect to the required databases and systems that the application uses.
These files can be generated using the Firebase console and attaching the project to it as explained in section 5.2.1 that describes all the information related to this service.

5.1 Tools used

5.1.1 Android Studio

The main tool used to develop this project was Android Studio. Android Studio [30] is the official Integrated Development Environment (IDE) for Android application development, based on IntelliJ IDEA. On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance productivity when building Android applications. Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

- Android application modules
- Library modules
- Google App Engine modules

The main project structure can be seen in the figure attached to the next page.

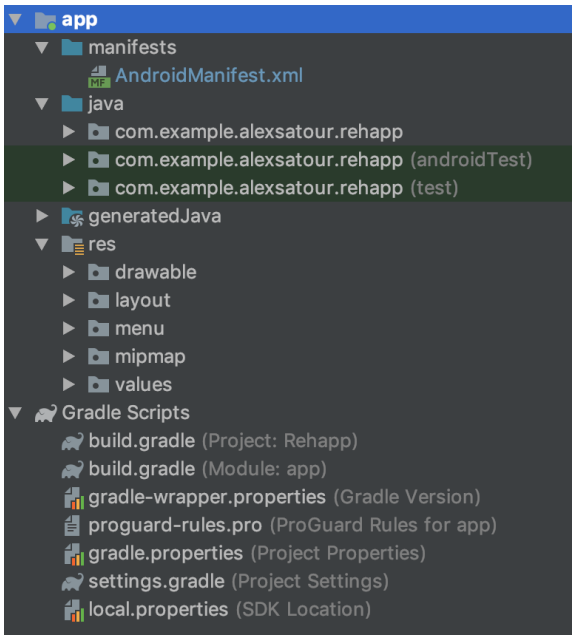

Figure 53: Android Studio project structure.

All the build files are visible at the top level under Gradle Scripts and each application module contains the following folders:

- **manifests**: Contains the AndroidManifest.xml file.
- **java**: Contains the Java source code files, including JUnit test code.
- **res**:Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

The manifest file[31] for an Android application is a resource file which contains all the details needed by the Android system about the application. It is a key file that works as a bridge between the Android developer and the android platform. It helps the developer to pass on functionality and requirements of our application to Android. The res folder[32] mainly contains the following folders:

- **color**: XML files that define a state list of colors.
- **drawable**: Bitmap files (.png, .9.png, .jpg, .gif) or XML files.
- **mipmap**: Drawable files for different launcher icon densities.
- **layout**: XML files that define a user interface layout.
- **menu**: XML files that define application menus, such as an Options Menu, Context Menu, or Sub Menu.
- **values**: XML files that contain simple values. The most important would be:
  - **arrays.xml**: Contains resource arrays.
  - **colours.xml**: Contains colour values.
  - **dimens.xml**: Contains the dimension values such as text sizes or layout dimensions.
  - **strings.xml**: Contains string values.
  - **styles.xml** : Contains the definitions of styles such as colour patterns or text sizes.

In addition, it is really important to provide alternate resources such as different image sizes for screens with different pixel densities, different string resources for each language that the application will be used or a specific layout to use in case the device is in portrait mode or landscape. Android Studio uses Gradle as the foundation of the build system, with more Android-specific capabilities provided by the Android plugin for Gradle. This build system runs as an integrated tool from the Android Studio menu, and independently from the command line.

5.2 Technologies Used

5.2.1 Firebase

In order to store all the information online and be able to retrieve it and interact with it, a cloud storage service was needed.
Many options were studied but the best one without a doubt is the service provided by Google named Firebase since it was created to solve needs like the one that exists in this project. Firebase[33] is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014. As of October 2018, the Firebase platform has 18 products, which are used by 1.5 million applications.
Firebase evolved from Envolve, a prior startup founded by James Tamplin and Andrew Lee in 2011. Envolve provided developers with an API that enables the integration of online chat functionality into their websites. After releasing the chat service, Tamplin and Lee found that it was being used to pass application data that weren't chat messages. Tamplin and Lee decided to separate the chat system and the real-time architecture that powered it. They founded Firebase as a separate company in September 2011 and it launched to the public in April 2012.
Firebase handles the backend online element for applications, allowing developers to focus on front-end UI and functionality. All this is done through a single SDK with easy-to-use APIs and integration into Android Studio. This removes the need to create an own server-side script using PHP and MySQL, or similar set-up. This is 'Backend as a Service' or 'BaaS'. It works with Android applications, iOS applications and web applications and it is free.
It offers different service to solve the needs of a project, they can all be seen classified based on the functionality they provide in the chart attached below:

Figure 54: Services provided by Firebase[34].

As we can see, there are many services offered by Firebase but only a few will be used in this project. The first one is Firebase Authentication. It provides backend services, easy-to-use SDKs, and ready-made UI libraries to authenticate users to the application. It offers multiple methods to authenticate, including email and password, third-party providers like Google or Facebook. Obviously, it will be used to manage the login process of the users of the application and to store the main data about users such as their name or their profile picture.
Firebase offers mainly two databases to store information. One is Firebase Realtime Database[35] and it is a cloud-hosted database. Data is stored as JSON and synchronised in real-time to every connected client. When a cross-platform application is built with iOS, Android, and JavaScript SDKs, all of the clients share one Real-time Database instance and automatically receive updates with the newest data. This will be used to store the additional information of a user that Firebase Authentication cannot store such as its type or the description of an injury.
The second database option would be Cloud Firestore. It[36] is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform. Like Firebase Real-time Database, it keeps data in sync across client applications through real-time listeners and offers offline support for mobile and web so developers can build responsive applications that work regardless of network latency or Internet connectivity. Cloud Firestore also offers seamless integration with other Firebase and Google Cloud Platform products, including Cloud Functions. It will be used to store information such as exercises or appointments since it is easier to perform queries against it.
The last service that will be used is Cloud Storage. It[37] is a powerful, simple, and cost-effective object storage service built for Google scale.
The Firebase SDKs for Cloud Storage add Google security to file uploads and downloads for Firebase applications, regardless of network quality. It can be used to store images, audio, video, or other user-generated content. In this project, it will be used mainly to store the user profile images and the visual resources used throughout the application such as the pictures attached to describe an exercise.
To be able to use it, a project has to be created on the Firebase dashboard. Once it is created, it has to be linked to the Android Studio project. This can be done using a tool provided by the IDE.

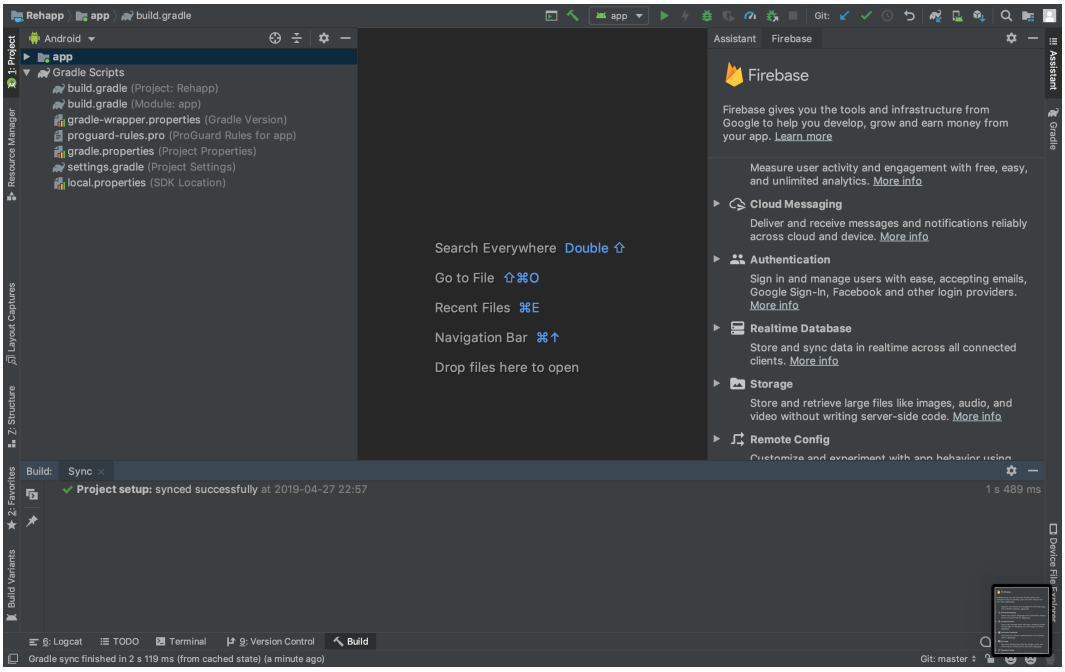

Figure 55: Tool provided by Android Studio to connect a project to Firebase.

Using this tool, all services can be easily set up following the instructions and steps assigned to each one.

Once all steps are done, the dashboard of the project should look like the screenshot attached below.

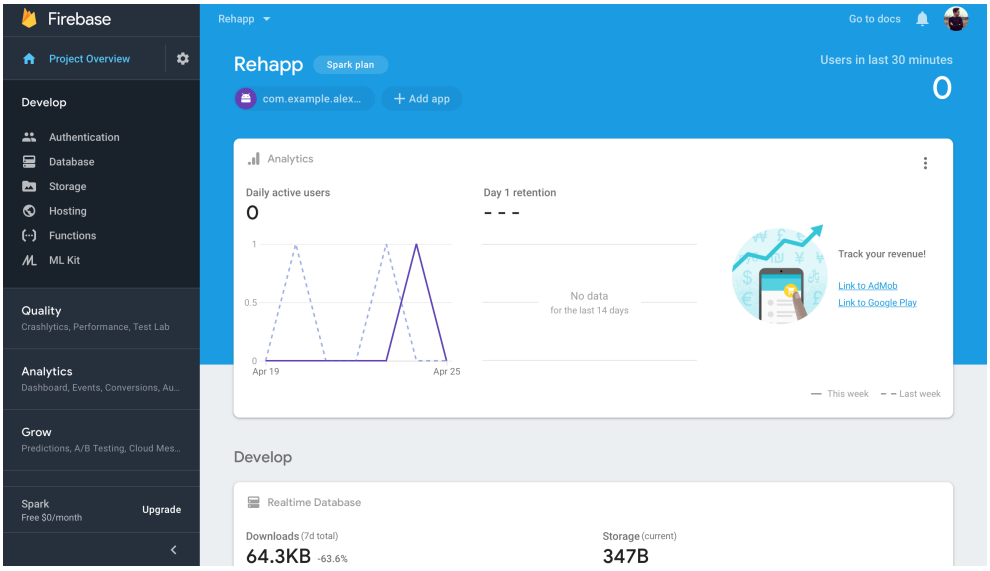

Figure 56: Firebase dashboard once it is linked to the Android Studio project.

Through this website, all services and settings can be established in order to have the services working as needed. Access permissions and functionalities can also be set up. In addition, all databases can be explored and managed very easily leading into great control of every aspect about them. In addition, it is very helpful to be able to see how data is stored in order to do it in the most effective way possible.

5.3 Development process

Once it was time to start developing the application it was really important to have in mind all the work that led into this phase and take it into account.
All the project was built in the logic order of screens, meaning that the first one to be implemented was the login screen and once it was working, the following ones were the ones that depended on it.
Obviously, some aspects appeared throughout the development process that were not realised before.
One of them was that an extra role apart from patients and medical crew member was necessary to create the profiles associated with medical crew members and to manage all the users. As a consequence, an extra role that would do the tasks of an administrator was created.
In addition, when implementing the screens and testing them on different devices, another issue appeared. It was the fact that this application would be used in many different devices and some might have a small screen, leading to having to pack as much information as possible and remove unnecessary elements. As a result, it was decided to remove the action bar in the screens that were not extremely necessary. Leaving it only in the screens that include some kind of list and it comes in handy to manage search and change the way the data is presented.
Once it was finished, the part of the project related to the functionalities ended up looking like the figure attached to the following page.

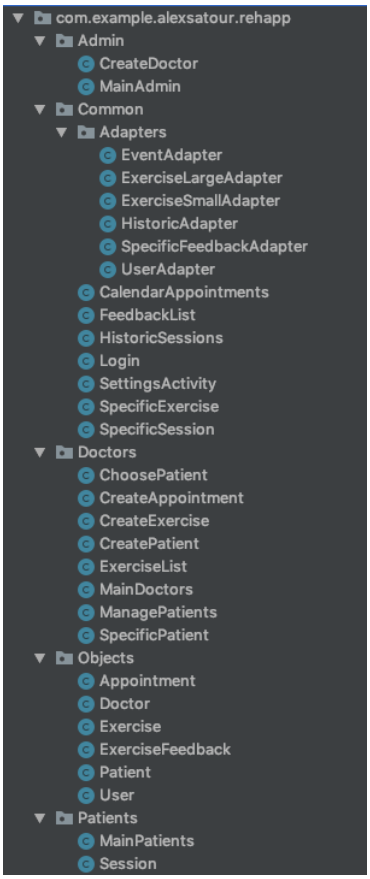

Figure 57: Project files when the development process ended.

It was structured in different packages in order to have all the different classes as structured as possible. The first one is the package related to the functionalities associated with an administrator.
The next package contains all the functionalities that were common in all users or could be used by multiple types of them in order to have the less duplicate code as possible. As it shows, there is another package inside named adapters that contains the classes helpful to show items on a list. This is why there are many of them depending on which kind of item they are associated with. All the other files inside the folder named common are activities that as said before, could be used by more than one kind of profile.
The third package contains the activities associated with the functionalities and screens related specifically to the members of the medical crew.
Next on, the package Objects contains classes that are helpful to represent some entities that are mainly stored on the database in order to be able to create them easily and upload them without having to structure the information manually.
Lastly, the folder named patients contains the activities related to the functionalities that patients need.

5.3.1 Libraries used

In order to provide the best user experience as possible and an appealing design, some external libraries were used. Its usage was crucial in order to provide some functionalities that might require a lot of work to implement by themselves.

Hourglass
Hourglass[38] is a Countdown timer that provides the feature to pause and resume the timer. The CountDown timer provided by Android has the issue that it can not be paused and resumed easily, which is solved by the functionalities provided by this library. It was really useful to implement the interactions with the timers when patients are working on a session since they must be able to stop it at any given time.

Android SwitchDateTime Picker
SwitchDateTime Picker[39] is a library that can be used to select a Date object in dialog with a DatePicker (Calendar) and a TimePicker (Clock) in the same UI. This was crucial to implement a very useful interface to select the specific date and time of an event without having the need of moving to an extra screen and only using a pop-up.

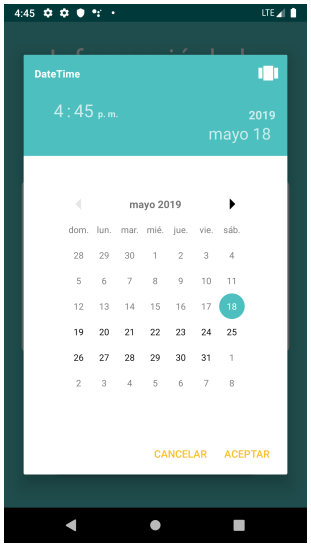

Figure 58: Interface to set a day.

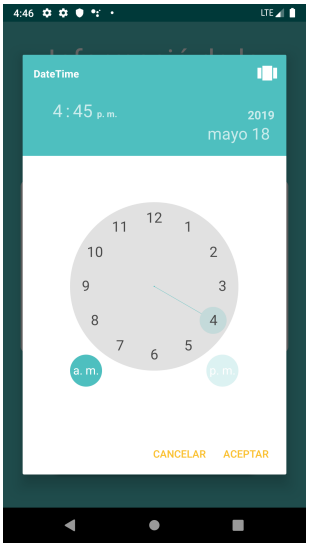

Figure 59: Interface to set an hour.

Smiley Rating
SmileyRating[40] is a simple rating bar for Android, it displays animated smileys as rating icon. It was very useful to be able to include an interactive way to let users leave their feedback on a certain exercise.

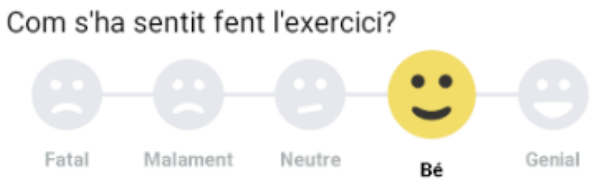

Figure 60: Smiley rating bar.

Text Field Boxes
This library[41] improves the text field boxes available by default giving them some design upgrades such as changing the title size when a user starts writing on them.

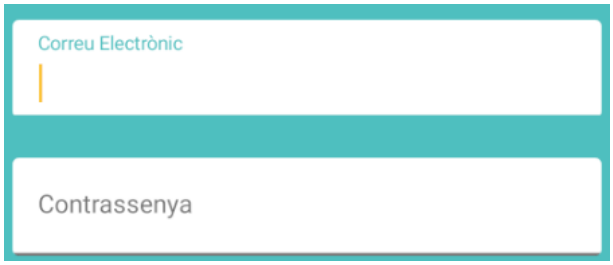

Figure 61: Text field boxes in their two different states.

Picasso
Picasso[42] is a powerful image downloading and caching library for Android. It was crucial to load all the pictures that are stored online only using a link and without having to control anything of the download process.

Circle ImageView
This library[43] came really in handy to create an appealing way of showing the profile pictures of users easily since it creates a circular image out of any picture.

Figure 62: Circular ImageView showing a profile picture.

Number Picker
Number picker[44] is a library based on android.widget.NumberPicker. As TextFieldBoxes library, it extends the capabilities of an already existing widget making it more customisable and visually appealing. It is used to let doctors choose the duration, resting time and repetitions when creating a new exercise. An example of the interface that it provides can be seen in figure 63 attached to the following page.

Figure 63: Interface using Number Pickers to set the duration, resting time and repetitions when creating a new exercise.

Compact Calendar View
CompactCalendarView[45] is a simple calendar view which provides scrolling between months. It's based on Java's Date and Calendar classes. It provides a simple API to query for dates and listeners for specific events. It was used in all the screens that needed a calendar in them due to all the functionalities that it offers.

Figure 64: Compact Calendar View used in the screen that shows the future appointments that a certain user has scheduled.

5.3.2 Providing alternative resources.

In order to make the application functional in as many aspects possible, many alternative resources were provided.
First and foremost, all text resources that appear throughout the application are externalised and properly translated to English, Spanish and Catalan since these are the three main languages that users will use.
In addition, the images used are available in all screen densities so that the best version can be selected in each situation, leading into faster loading times since a device with a small screen does not need to load an image with a resolution higher than the device is able to show.
Most of the layouts have their space distributed using weights and not final values. This will guarantee that space is distributed equally in the different screen sizes available and that all elements will be shown.

5.3.3 Managing users

Since there are many types of users, a way to manage them had to be created. Since these profiles follow a hierarchy, it will be used to establish which profile can create a new member. Only administrators will be able to create new profiles associated with medical crew members just like only medical staff will be able to create new patient profiles.
Since this application is not open to the public, each new member has to be created from the device of a person that is above them in the hierarchy. This way, unwanted registrations will be completely avoided. Just as creating new profiles, deleting them will work the same way.
To be able to store the data related to the user and identify the different type to provide them with their specific functionalities the class User was created.

```
public class User {
    private int type;
    private String name, image;

    public User() {
    }

    public User(int type, String image, String name) {
        this.type = type;
        this.image = image;
        this.name = name;
    }

    public int getType() {
        return type;
    }

    public String getImage() {
        return image;
    }

    public String getName() {
        return name;
    }
}
```

Figure 65: Class used to represent an user.

An object of type user will store the information related to a user in order to be able to store it on the Firebase Database and retrieve it when necessary and be capable to list users easily. In addition, the name and image of a user will also be associated with its Firebase Authentication instance so that we can swiftly retrieve information about the user that is using the application.
To be able to differentiate each kind of user, the field type was created. Its values can be:

- **0:** Represents a patient.
- **1:** Represents a member of the medical crew.
- **2:** Represents an admin.
- **3:** Represents an user that has been deleted.

The fourth user type was created to represent a user that is no longer authorised to log in.
Since a patient needed some extra fields of information, a subclass was created in order to be able to store them. Its structure can be seen in the figure 66 attached to the following page.

```java
public class Patient extends User {
    private String les, tel;

    public Patient(String les, String tel, String name, String picture) {
        super( type: 0, picture, name);
        this.les = les;
        this.tel = tel;
    }

    public String getLes() { return les; }

    public String getTel() { return tel; }
}
```

Figure 66: Class used to represent a patient.

As we can see, it establishes the type to 0 and adds the fields that will store their phone number and injury. In addition, a class named Doctors was created but it only calls the constructor defining its type as 1.
Administrators would be created manually using the Firebase interface since it is not usual to create them and there will only be a few that would have access to the Firebase database.
The login process works as shown in the figure attached below.

```java
private void login() {
    FirebaseDatabase.getInstance().getReference( path: "Users")
            .child(Objects.requireNonNull(FirebaseAuth.getInstance().getCurrentUser()).getUid()).addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            pd.hide();
            pd.dismiss();
            if (dataSnapshot.child("type").getValue(Long.class) == 0) {
                startActivity(new Intent( packageContext: Login.this, MainPatients.class));
                finish();
            } else if (dataSnapshot.child("type").getValue(Long.class) == 1) {
                startActivity(new Intent( packageContext: Login.this, MainDoctors.class));
                finish();
            } else if (dataSnapshot.child("type").getValue(Long.class) == 3) {
                Toast.makeText( context: Login.this, "El usuari introduït ha estat eliminat.", Toast.LENGTH_SHORT).show();
                FirebaseAuth.getInstance().signOut();
            } else {
                startActivity(new Intent( packageContext: Login.this, MainAdmin.class));
                finish();
            }
            loadingProgressBar.setVisibility(GONE);
        }

        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {
            loadingProgressBar.setVisibility(GONE);
        }
    });
}
```

Figure 67: Method used to identify which kind of user is trying to log in.

As seen in the figure, each user has an instance in the Firebase database associated with their user ID. This information is retrieved and checked to be able to redirect each user to their respective main screen. When a user with the privileges required proceeds to delete a profile, all the information related to the user that will be removed will be saved and the only thing that will change is the value associated to this user type. That is because it can be useful to keep storing the information related to them since it could be needed in the patient. In addition, accidental profile deletion could be reverted.
All of the activities shared have been implemented only once and designed to properly adapt to offer the specific functionalities to each type of user and avoid any duplicate code.

5.3.4 Working with online data

The database used to store the information that users need is Firebase's Firestore. It was chosen due to its scalability and how easy it is to upload data and retrieve it as well as making queries.

The database ended up having four collections that store all the information:

1. **Calendar:** Contains all the information related to appointments such as the doctor and patient ID or the date of the appointment.
2. **Exercises:** Stores all the exercises available as well as the information related to each one of them.
3. **Historic:** Keeps all the data about the sessions done by patients. Mainly the date they did them and the feedback information they left on each exercise as well as an average.
4. **Sessions:** Stores the exercises that the session of each specific patient includes.

Retrieving and saving data is always done the same way but changing small details such as the collection or the document that stores the data. The first step to do would be getting the reference to the desired collection or document.
Once the reference is obtained, if the operation needed is related to saving data, the method used is `add(Object o)`. This method needs the object to be stored as a parameter. Once this is done, a method named `addOnCompleteListener` is used to know the result of the operation and act consequently. Attached below we can see a figure that shows the code that creates an appointment and uploads it.

```
Appointment a = new Appointment(new Event(getResources().getColor(R.color.colorAccent), fdate.getTime()), FirebaseAuth.getInstance().getCurrentUser().getUid(), patID)
FirebaseFirestore.getInstance().collection( collectionPath: "Calendar") CollectionReference
        .add(a) Task<DocumentReference>
        .addOnSuccessListener((OnSuccessListener) (documentReference) → {
                Toast.makeText( context: CreateAppointment.this, "S'ha guardat la sessió correctament!.", Toast.LENGTH_SHORT).show();

        }) Task<DocumentReference>
        .addOnFailureListener((e) → {
                Toast.makeText( context: CreateAppointment.this, "No s'ha pogut guardar la sessió! Torni a intentar-ho més...", Toast.LENGTH_SHORT).show();
        });
```

Figure 68: Creation of an appointment and storing it in Firestore.

In the other hand, if we need to retrieve information the method `get()` is the one to use. In addition and if needed, queries can be added before this method in order to filter the data retrieved. Just as with the method associated with storing information, the method `addOnCompleteListener` should also be used. In this case, it will be necessary to check if the task did not fail and then, iterate the list of DocumentSnapshots and convert them.
A DocumentSnapshot is simply the representation that an object has in the Firestore Database, as said before they must be converted to the Object they actually are when retrieved. This can be seen in the figure attached below that obtains the list of events associated with a user.

```
private void getAppointments(String usr, String name) {
    pd.show();
    FirebaseFirestore.getInstance().collection( collectionPath: "Calendar") CollectionReference
            .whereEqualTo(usr, name) Query
            .get() Task<QuerySnapshot>
            .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
                @Override
                public void onComplete(@NonNull Task<QuerySnapshot> task) {
                    if (task.isSuccessful()) {
                        List<DocumentSnapshot> l = Objects.requireNonNull(task.getResult()).getDocuments();
                        for (DocumentSnapshot document : l) {
                            //Convert to object and add it to a list
                        }
                    }
                });
}
```

Figure 69: Method that retrieves the events associated to an user from Firestore.

Of course, the code seen above has been simplified to include only the important parts of the process. In addition, it can also be seen how to perform a query to filter the data that will be retrieved.

Since users might need to interact with the downloaded data, many activities also create a HashMap to store and link each specific item with the ID that represents it in the database in order to be able to modify, delete or obtain more information about it.

5.3.5 Designing the screens

Android uses two files that compose what is perceived as a screen. The functional part named Activity that can be found in a file written in Java and the design part that is found on a file written in XML. The visual representation of a screen is also called layout and is stored in the res folder that contains all the resources.
All screens were designed using mainly two layouts provided: Linear Layout and Relative Layout.
Linear Layout[46] is a type of layout that arranges other views either horizontally in a single column or vertically in a single row. In addition, it lets developers assign a weight to each item contained to specify how linear layout divides remaining space amongst the views it contains It makes it really easy to distribute elements in the screen and adapt it to numerous screen sizes. Relative Layout[47] is a layout that displays child views in relative positions. The position of each view can be specified as relative to sibling elements (such as to the left of or below another view) or in positions relative to the parent Relative Layout area (such as aligned to the bottom, left or centre). It is a very powerful utility for designing a user interface because it can eliminate nested view groups and keep your layout hierarchy flat, which improves performance. Several nested Linear Layout groups can be replaced with a single Relative Layout.
As said before, it was really important to avoid any duplicate code. As a consequence, many layouts are designed to adapt to each kind of user and adapt to each one depending on the functionalities they need. This was achieved hiding and showing elements as needed and distributing the space of the screen smartly.
Some examples of changing layouts are attached below.

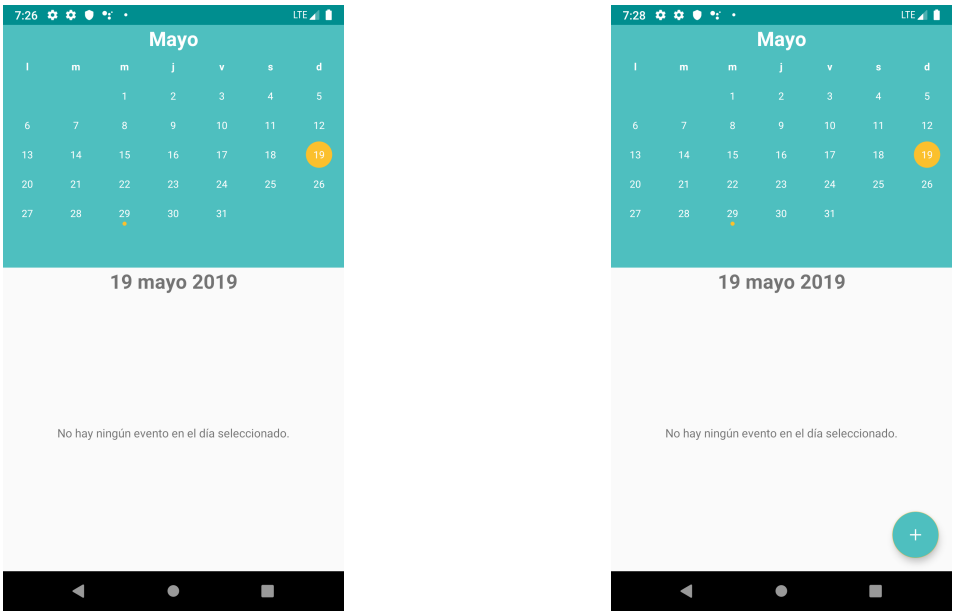

Figure 70: Versions of the screen that shows future appointments depending on which type of user is seeing it.

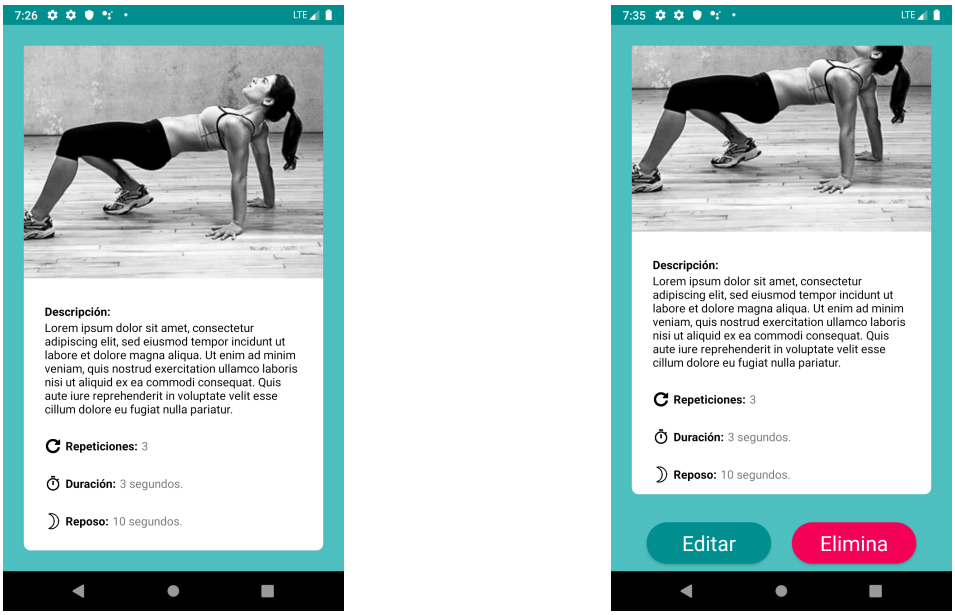

Figure 71: Different versions of the screen that shows the details of an exercise depending on which type of user is seeing it.

On both figures, the screenshot seen in the left is the one that corresponds to patients and the image in the right, the one belonging to the medical crew version.

5.3.6 Showing lists of information

In order to display lists of data in the most efficient way possible, a widget named Recycler View was used and the View Holder pattern was applied. RecyclerView[48] is able to detect data set changes in batches during a layout calculation. This saves LayoutManager from tracking adapter changes to calculate animations. It also helps with performance because all view bindings happen at the same time and unnecessary bindings are avoided. In addition, data may be placed in a cache for later reuse to display the same type of data again later. This can drastically improve performance by skipping initial layout inflation or construction.
ViewHolder[49] is a design pattern which can be applied when using a custom adapter. Every time the adapter calls getView() method, the findViewById() method is also called. This is very intensive work for the mobile CPU and so affects the performance of the application and the battery consumption increases To avoid this, ViewHolder is used. A ViewHolder holds the reference to the id of the view resource and calls to the resource will not be required. Thus the performance of the application increases.
It is really important to understand that an adapter is an element responsible for providing views that represent items in a data set. As a consequence and as seen in Figure 57 an adapter for each kind of data had to be created. In addition, two adapters were developed in order to be able to show the exercises with a big image that would show how to do it and with only the name in order to have a more compact way to display them. A user can switch the way of seeing the exercises using the button located at the top right corner of the screen as it can be seen in the figures attached to the following page.

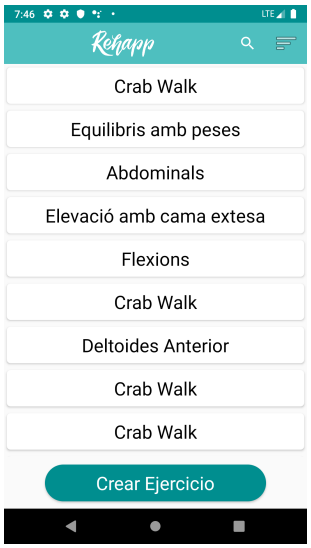

Figure 72: Compact exercise list.

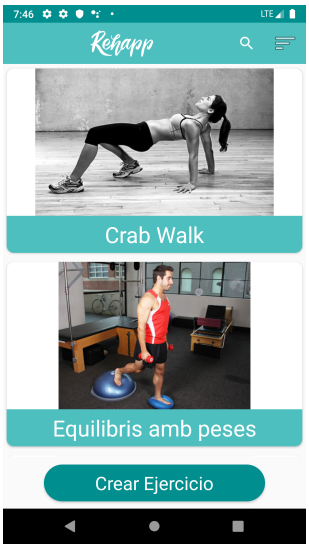

Figure 73: Expanded exercise list.

Furthermore, a SearchView Widget was implemented to let users filter data and get the item they want easily. This was achieved implementing the interface `OnQueryTextListener` and overriding its methods. In addition, a function named filter was created in each adapter in order to be able to filter each type of data depending on their most important fields.

5.3.7 Displaying appointments and past sessions

To implement these two crucial functionalities, two activities were implemented but only one layout was needed since they require the same design.
Both activities do almost the same but they differ on the collection from which they retrieve the information since the data related to the future appointments can be found in the collection Calendar and the one associated to the feedback left is stored in the collection named History.
Both versions use the same pattern but using different resources. They obtain the events and load them in the calendar as well as storing the data associated that contains the information about an appointment or a session.
As seen in Figures 76 and 77, the library used to display the calendar allowed a more aesthetically pleasing calendar than the one provided by default and used in the prototype. Furthermore, it offered a way to represent the days that have some kind of event showing a dot under each date that has something scheduled.
Since it is a library used to display appointments, it also provides a type of object named Event that makes the process of representing a rendezvous very easy. This object is able to store the colour used to represent the appointment as well as the specific date scheduled and extra data associated to it.
Once the information is downloaded, it is loaded into each adapter in order to be able to display the data properly.
The results of these tweaks can be seen in Figures 76 and 77 attached to the following page.

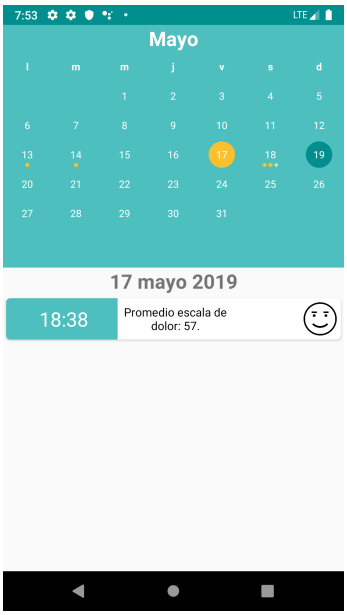

Figure 74: Screen showing past sessions.

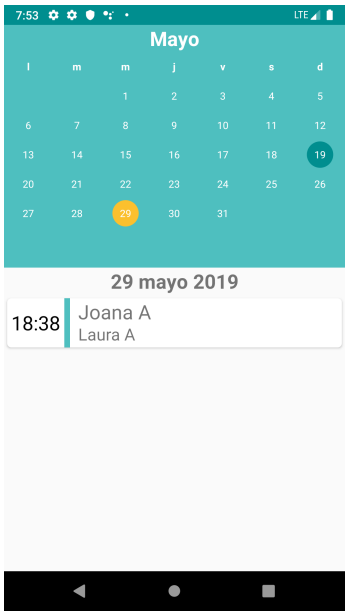

Figure 75: Screen showing future appointments.

5.3.8 Developing a session

This was one of the most challenging activity to develop because it was composed of 3 states. The three states would be the waiting screen that shows a countdown until it is time to start an exercise, the exercise screen itself and the screen used by patients to leave the feedback. Of course, the countdown state will also be shown whilst waiting for the resting time between each repetition of an exercise is over. This three-screen cycle will repeat itself for each exercise until the session finishes.
The screens representing each state can be seen in the figures attached below.

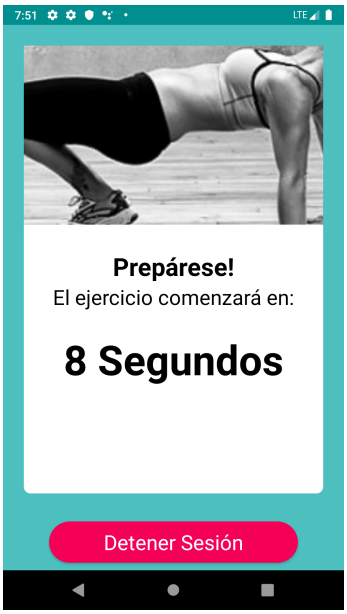

Figure 76: Countdown waiting screen.

Figure 77: Exercise information screen.

Figure 78: Feedback input screen.

Many options such as creating a different activity for each state were considered. Nevertheless, the chosen one was to have a single activity and a single layout associated with it. In this way, the main layout has three sub-layouts and the visibility of each one is controlled programmatically. The main advantage is that there are no waiting times when switching between screens and no information has to be reloaded. In addition, no data has to be transferred between them.

5.3.9 Managing user preferences

In order to let users set their preferences, many resources provided by AndroidX Preference Library were used. This library[50] manages the user interface and interacts with storage so that developers define only the individual settings that the user can configure. The library comes with a Material theme that provides a consistent user experience across devices and OS versions.
A Preference is the basic building block of the Preference Library. A settings screen contains a Preference hierarchy defined as an XML resource. When building a hierarchy, each Preference should have a unique key that can be accessed at any time to retrieve the value associated with it. In this project, the preferences created are:

1. **Name:** Lets users change their display name.
2. **Password:** Lets users reset their passwords.
3. **Automatic login:** Keeps the authentication state in order to be able to log in without having to enter the credentials every time.
4. **About:** Shows user the information about the development of the application.

Obviously, many of these fields are also stored in the cloud and they must be updated when changed. This can be done setting a listener that updates every instance of the field modified. To inflate a hierarchy from an XML attribute, a PreferenceFragmentCompat has to be created, the `onCreatePreferences()` method overrided, and an XML resource has to be provided to be inflated.
In addition, a section was designed to let users replace their profile picture above the preference section of the settings screen.

6 Conclusions

Once the application development stage was finished, it was time to evaluate the work done and obtain some conclusions.
This was, without a doubt, the most challenging project I have ever faced due to its complexity and the amount of work it required. In addition, it was not easy to be able to find a way to balance this project and taking full-time classes as well as having to study and work on assignments required.
As a result, the most difficult part of developing this project was finding a way to time each stage. The way that ended up working up the best was setting deadlines for each phase because otherwise, it was very easy to keep working on a phase and forget about the global project.
Attached below, we can see both the timings scheduled at the beginning of the project and the real timings and deadlines that ended up happening.

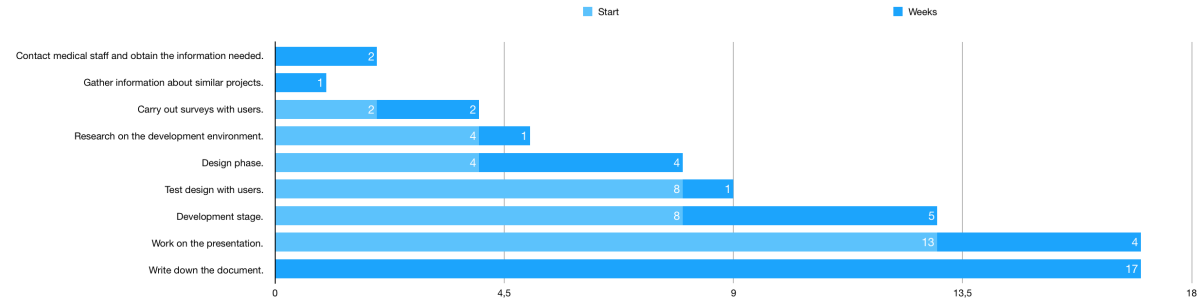

Figure 79: Gantt chart showing the expected timings of the project.

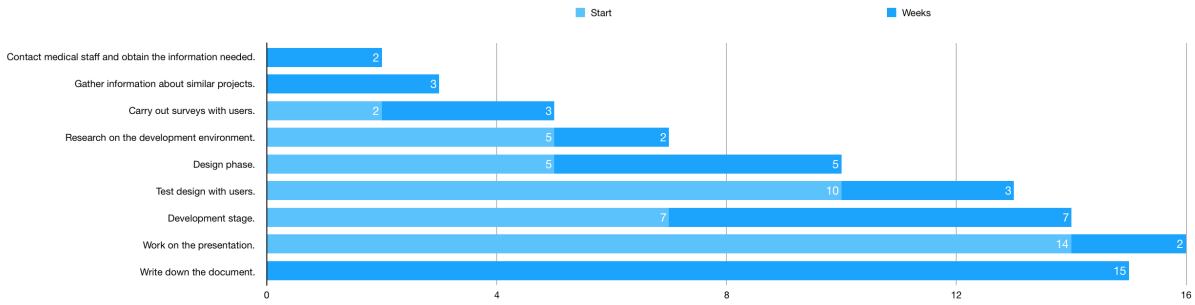

Figure 80: Gantt chart showing the actual timings of the project.

As shown in charts, many of the phases had to be delayed due to external reasons such as doctor's schedule. In spite of that, the project lasted a week less than expected. The reason behind it is that the delays mentioned before created a sense of having to work more to get to the expected results although there was no need in rushing.
In addition, all the goals set and my vision of the project have been achieved successfully. The development of this project has also taught me how to deal with unexpected circumstances and how to work them out in order to achieve the goals set.
Throughout the development of this project, I have been able to acquire lots of knowledge in the many areas explored to be able to build the final application. Having to work by myself on areas that were unknown pushed me to explore them in order to take every decision for a reason and be able to develop the best project I could.

Developing this project has shown me how crucial it is to work equally as hard on every phase of a project. Working like that and receiving external feedback or working with people who were not working on the project itself reassured that each stage was well done and made that step that had to be implemented later on, was built on solid ground. In addition, it made it really easy to work on each phase since the work that had to be done was delimited. Furthermore, the stages before defined what had to be done.
It has also helped me to see new perspectives and encouraged me to work with people specialised in different areas in order to obtain the best work possible. This led me to learn from professionals of fields that I had never explored and understanding how to manage new points of view and include them on a personal project.

7 Future work

Obviously, this project could be developed much further than it is at the last stage described in this document. This is because the time to work on it is limited and the main point behind developing this project was to be able to build an application whilst studying and learning from each phase that leads up to that stage. Nevertheless, there are many points that could be worked on to improve the project.
The first one could be to perform a test process with users to evaluate the application developed since it has many changes that should be tested before the application is released in order to assure that everything works as expected.
Due to timing issues and limitations of the developing tools, there are some improvements that could be done to provide the best user experience possible.
The first one could be providing multiple translations of the descriptions of the exercises since they are only available in one language in the current implementation.
Another way to improve the user experience would be adding a notification system to let users know when an appointment has been rescheduled or cancelled. This way, the need for checking it manually would disappear.
A functionality that could be added despite medical crew stated that was not crucial would be letting doctors edit an exercise to be more specific to the treatment of a patient. The way exercises of a session are stored in the Database was created having in mind that this functionality might be added. The implementation of this capability would be very easy since each patient has its own copy of the exercise data.
There is a small bug that should be fixed if the project is worked further on. It is to avoid that a user logs out when they create another profile. This could be fixed using an Admin SDK to manage the creation of new profiles
It could also be helpful to let doctors add videos to an exercise description so they can be more self-explanatory and not rely only on a text describing how to do the exercise.
An additional functionality that would improve the communication process even though patients and doctors see each other very often would be adding messaging capabilities so that they can be able to talk to each other about many questions such as doubts about the treatment or discuss on changing a scheduled appointment.
Creating a whole section to let doctors see deleted patients data and delete them permanently would be really easy to implement and would be a great functionality to manage their information and to remove unnecessary data from the database.
Lastly, the point that could improve the most the project would be adding the capability of measuring each exercise in order to track precisely the evolution of patients and having more information than the feedback left by them. This could be done using systems similar to the ones explained at the State of art section at the beginning of this project or even using the sensors available on smartphones.

References

[1] What is Agile?, Retrieved: March 6th, 2019, https://www.agilealliance.org/agile101/.

[2] What is User Centered Design?, Retrieved: March 11th, 2019, https://www.interaction-design.org/literature/topics/user-centered-design.

[3] Android (operating system), Retrieved: March 16th, 2019, https://en.wikipedia.org/wiki/Android_(operating_system).

[4] android-stack_2x.png, Retrieved: March 16th, 2019, https://developer.android.com/guide/platform/images/android-stack_2x.png

[5] Android version history, Retrieved: March 16th, 2019, https://en.wikipedia.org/wiki/Android_version_history

[6] Mobile & Tablet Android Version Market Share Worldwide, Retrieved: March 16th, 2019, http://gs.statcounter.com/os-version-market-share/android/mobile-tablet/worldwide

[7] Platform Architecture | Android Developers, Retrieved: March 16th, 2019, https://developer.android.com/guide/platform

[8] The 13 Best New Features in Android 9.0 Pie, Retrieved: March 16th, 2019, https://www.makeuseof.com/tag/new-features-android-pie/

[9] The 9 best new features in Android 9 Pie... and how to get it, Retrieved: March 16th, 2019, https://www.stuff.tv/features/9-best-new-features-in-android-9-pie-and-how-get-it

[10] *¿Qué es Toyra?*, Retrieved: March 21st, 2019, http://www.toyra.org/que-toyra/.

[11] 013_toyra_rehabilitation.48bd3dff.jpg, Retrieved: March 21st, 2019, https://fitness-gaming.com/assets/image-cache/Markets/Health_and_Rehab/1103_toyra_introduces_virtual_reality_to_occupational_therapy/013_toyra_rehabilitation.48bd3dff.jpg.

[12] DyCare: clinical assessment by wearable sensors solutions, Retrieved: March 21st, 2019, https://dycare.com.

[13] DyCare: ReHub, Retrieved: March 21st, 2019, https://dycare.com/rehub/.

[14] DyCare: Lynx, Retrieved: March 21st, 2019, https://dycare.com/lynx/.

[15] Backgrounds.jpg, Retrieved: March 21st, 2019, https://www.dycare.com/wp-content/uploads/2019/02/Backgrounds.jpg.

[16] *EXPERIÈNCIES EN TIC I SALUT*, Retrieved: March 21st, 2019, http://www.ticsalut.cat/observatori/innova-tic-salut/68/rehabitic.

[17] reyhabitic.jpg, Retrieved: March 21st, 2019, http://1.bp.blogspot.com/_WiqMAWN19Js/TD2Cxhf7ypI/AAAAAAAAHxU/s2w7AtpeqmY/s1600/reyhabitic.jpg.

[18] ITER - *Instituto Tecnológico y de Energías Renovables*, S.A., Retrieved: May 21st, 2019, https://www.iter.es/portfolio-items/tango-h/?lang=en.

[19] tangoh_03.png, Retrieved: May 21st, 2019, https://www.iter.es/wp-content/uploads/2016/01/tangoh_03.png.

[20] Projects, Retrieved: May 21st, 2019, http://www.penichet.net/projects-2.

[21] EME-REHAB.png, Retrieved: May 21st, 2019, http://www.penichet.net/wp-content/uploads/2016/01/EME-REHAB.png.

[22] esquema-editor-sivire.png, Retrieved: May 21st, 2019, http://www.penichet.net/wp-content/uploads/2016/01/esquema-editor-sivire.jpg.

[23] When and How to Use Ethnographic Research, Retrieved: March 28th, 2019, https://www.spotless.co.uk/insights/ethnography-when-and-how/.

[24] Adobe XD, Retrieved: April 5th, 2019, https://en.wikipedia.org/wiki/Adobe_XD.

[25] Color, psychology and design, Retrieved: April 8th, 2019, https://uxplanet.org/how-color-can-effect-emotion-ccab0431b1d.

[26] Most liked colors by men and women, Retrieved: April 8th, 2019, https://cdn-images-1.medium.com/max/1600/1*LcVqPav9MwJ8SmCQFfX_OQ.png

[27] Most disliked colors by men and women, Retrieved: April 8th, 2019, https://cdn-images-1.medium.com/max/1600/1*LcVqPav9MwJ8SmCQFfX_OQ.png

[28] System Usability Scale (SUS), Retrieved: April 14th, 2019, https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html

[29] alexsat97/TFG-Rehapp, Retrieved: May 24th, 2019, https://github.com/alexsat97/TFG-Rehapp

[30] Meet Android Studio, Retrieved: April 22nd, 2019, https://developer.android.com/studio/intro

[31] Android Manifest, Retrieved: April 22nd, 2019, https://javapapers.com/android/android-manifest/

[32] App resources overview, Retrieved: April 22nd, 2019, https://developer.android.com/guide/topics/resources/providing-resources

[33] Firebase, Retrieved: April 29th, 2019, https://en.wikipedia.org/wiki/Firebase

[34] Chart of services provided by Firebase, Retrieved: April 29th, 2019, https://cdn-images-1.medium.com/max/1600/1*rIDMNpRx8HnnY5mhhbXLuA.png

[35] Firebase Realtime Database, Retrieved: April 29th, 2019, https://firebase.google.com/docs/database/

[36] Cloud Firestore, Retrieved: April 29th, 2019, https://firebase.google.com/docs/firestore/

[37] Cloud Storage, Retrieved: April 29th, 2019, https://firebase.google.com/docs/storage/

[38] Hourglass, Retrieved: May 13th, 2019, https://github.com/groverankush/Hourglass

[39] Android-SwitchDateTimePicker, Retrieved: May 13th, 2019, https://github.com/Kunzisoft/Android-SwitchDateTimePicker

[40] SmileyRating, Retrieved: May 13th, 2019, https://github.com/sujithkanna/SmileyRating

[41] TextFieldBoxes, Retrieved: May 13th, 2019, https://github.com/HITGIF/TextFieldBoxes

[42] Picasso, Retrieved: May 13th, 2019, https://github.com/square/picasso

[43] CircleImageView, Retrieved: May 13th, 2019, https://github.com/hdodenhof/CircleImageView

[44] NumberPicker, Retrieved: May 13th, 2019, https://github.com/ShawnLin013/NumberPicker

[45] CompactCalendarView, Retrieved: May 13th, 2019, https://github.com/SundeepK/CompactCalendarView

[46] Linear Layout | Android Developers, Retrieved: May 15th, 2019, https://developer.android.com/reference/android/widget/LinearLayout

[47] Relative Layout | Android Developers, Retrieved: May 15th, 2019,
https://developer.android.com/guide/topics/ui/layout/relative

[48] RecyclerView | Android Developers, Retrieved: May 15th, 2019,
https://developer.android.com/reference/android/support/v7/widget/RecyclerView

[49] Android Applications: What is ViewHolder in Android's Adapter class?, Retrieved: May 15th, 2019,
https://www.quora.com/Android-Applications-What-is-ViewHolder-in-Androids-Adapter-class

[50] Settings | Android Developers, Retrieved: May 15th, 2019,
https://developer.android.com/guide/topics/ui/settings